Clustering and Visualization of Network Security-Related Data using Elastic Stack

Vladimir Ciric, Marija Milosevic, Luka Mladenovic, Ivan Milentijevic

***Abstract*—Security concerns and economic losses caused by network attacks have inspired extensive network security research with active study fields that include data collection, analysis, and visualization. Visualization can help analysts to efficiently detect unusual behavior patterns in a vast amount of data, which should result in a prompt response to a potential security threat. However, the majority of the research papers suggest custom visualization solutions for the proposed analysis techniques rather than using available and well-adopted data pipelines that can further support the analysis. In this paper, we propose a system architecture for clustering, visualization, and computer-assisted network security analysis based on an open-source Elastic Stack. We extended the data pipeline in order to enable data clustering prior to visualization and employed visual mapping techniques to filter data and successfully spot the network assaults. A case study that demonstrates the effectiveness of the proposed solution is given in the example of a port scan attack.**

***Index Terms*—Network security, data clustering, data visualisation**

I. Introduction

Despite decades of research and a lot of available security products, the Internet has become more and more dangerous. Network penetration usually occurs in stages, and the only way to stay ahead of new vulnerabilities and attacks is through agile detection and fast response. However, accurate detection is a key component missing in most networks, and without proper and accurate alerts, administrators are literally ”flying blind” [1].

To obtain accurate security alerts, two key components are required: (1) a large amount of security-related data, including system logs, netflows, and reports from specialized security components; and (2) advanced algorithms for data processing and selecting relevant information. The data processing algorithms should help security analysts select relevant information, identify potential threats, and respond to them in a proper manner [2].

Security data analytics methods can be classified into three main categories: statistical, machine learning, and knowledge-based [2], [3]. A statistical inference is applied to calculate an anomaly score, which is then used to generate an alarm. The goal of machine learning is to create an explicit or implicit model of analyzed patterns, while knowledge-based approaches investigate network or host events for the existence of known attack scenarios by matching them with predetermined attack patterns [2].

Data clustering is one of the most common statistical approaches. Cluster analysis, often known as clustering, is the task of arranging a set of items so that objects in the same group (cluster) are more similar, in some aspects, to those in other clusters. In network security, clustering of collected network traffic data and relevant alerts can provide insight into relationships among them, allowing analysts to identify unusual patterns [4].

Data visualization has the potential to significantly improve network security analytics [5]. The analyst inspects summarized data through visual elements rather than exploring it in raw, usually textual form [6]. Visualization methods transform inflexible network anomaly data into graphical representations and provide interactive analysis modes to enhance human perception [5], [7]. However, the majority of the proposed solutions are designed using custom-built visualization systems built around a particular analysis method [5]. This leaves the security analyst with the problem of searching for additional analysis and data filtration tools once the anomaly is identified, which usually results in a complicated heterogenous environment for analysis [4].

In this paper, we extend the well-known and well-established data pipeline of open-source Elastic Stack in order to cluster security-related data prior to visualization. The proposed architecture enables clustering and visualization, followed by data analysis using native Elastic Stack capabilities. The data clustering is implemented using hierarchical clustering. A case study that demonstrates the effectiveness of the proposed solution in the example of a port-scan attack is given.

The paper is organized as follows: Section 2 covers the related work. In Section 3, the proposed architecture is presented. In Section 4, we give the case study, while in Section 5 concluding remarks are given.

II. Related work

Data clustering is one of the most common security analytic methods that has been successfully employed in cybersecurity in order to detect anomalies in network behavior and draw security analysts' attention to potential attack [2]. Along with

Vladimir Ciric is with the Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia (e-mail: vladimir.ciric@elfak.ni.ac.rs), (https://orcid.org/0000-0002-1442-7959)

Marija Milosevic is with the Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia (e-mail: marija.milosevic@elfak.ni.ac.rs),(https://orcid.org/0000-0002-3658-0292)

Luka Mladenovic is with the Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia (e-mail: luka.mladenovic@elfak.rs)

Ivan Milentijevic is with the Faculty of Electronic Engineering, University of Nis, Aleksandra Medvedeva 14, 18000 Nis, Serbia (e-mail: ivan.milentijevic@elfak.ni.ac.rs)

 ISBN 978-86-7466-970-9

the development of data analytic methods, considerable effort has been put into the development of visualization techniques [5]. Visualization transforms inflexible network anomaly data into graphical representations and provides interactive analysis modes to enhance human perception [7]. The conventional techniques in security literature can be categorized into analysis and visualization of network traffic [8], and analysis and visualization of security events [8], [9].

Zhou et al. [8] proposed network traffic analysis that transforms network data from temporal space to visual clustering space based on the entropy of IP address and port distributions. The authors showed that the entropy reflects the randomness of the hosts involved in network activities, from which conclusions on the network's behavior can be drawn. PerCIVAL is a visual analytics environment proposed in [10] that helps users understand network security status and monitor security events. The implemented tool gives a complex visualization of network topology overlayed with a potential attack graph. In [4] the authors addressed the dissimilarity of flow ensembles with regard to flow duration, density, and distribution, creating clusters of ensembles based on similar network behavior patterns. The visualization of clusters is performed in the form of a similarity matrix, followed by the visualization of temporal network flow data. VISNU is a visualization system proposed in [6] that combines the security events from multiple organizations for more effective analysis. The system helps find organizations with different event trends than others by visualizing the events in 3D space. The authors in [11] proposed Principal Component Analysis that provides a factorization of network traffic data. The goal is to find the subspace of maximum variance in the variable space. The designed visualization reflects the proposed method.

Reviewing the literature, it can be concluded that the majority of the proposed tools highly rely on the specific techniques and require the analyst to be familiar with complex methods used for data analytic [6], [10], [11]. Larger organizations, on the other hand, typically use the ELK based architecture (Elastic Search Engine, LogStash, and Kibana) [12], [13]. Elastic Stack is an open source engine that allows users to search, analyze and visualize data in real-time [13].

For example, in [12] Elastic Stack security architecture for securing IoT networks in smart cities is proposed. This system employs ELK endpoint security, with Beats for data shipping and Kibana for data visualization. It is designed to address the challenges relevant to the IoT. Similarly, the goal of the research published in [14] is to examine how well the Elastic stack tool can be used in threat hunting and compare it to other tools. It is shown that the Elastic Stack can be successful in identifying threats/security events while also being cost-effective.

In this paper, we extend the ELK architecture with a component for data clustering. The extension component implemented in this paper will implement agglomerative hierarchical clustering of the provided network data. Although the component will be designed to handle various types of data, we will focus on netflow and snort alert data in a similar manner as in [4], allowing visualization and further analysis in the ELK environment.

III. ELK BASED SYSTEM ARCHITECTURE FOR SECURITY DATA CLUSTERING

A. Architecture

The Elastic Stack is an open-source engine capable of importing data reliably from any source and in any format to search, analyze, and visualize it in near real-time [13]. It consists of four main components: ElasticSearch, Beats, Logstash and Kibana. ElasticSearch is built on top of Apache Lucene and Logstash is a server-side data processing engine that gathers data from different sources, transforms it and then sends to an arbitrary output. Beats are lightweight data shippers that are installed on servers to capture operational data of multiple types. Kibana is the analytics and visualization platform designed to search, view and interact with data stored in Elasticsearch indices [13].

Fig. 1 shows the four Elastic Search components where we propose the extension of the data pipeline with the configurable clustering of network traffic and alert data.

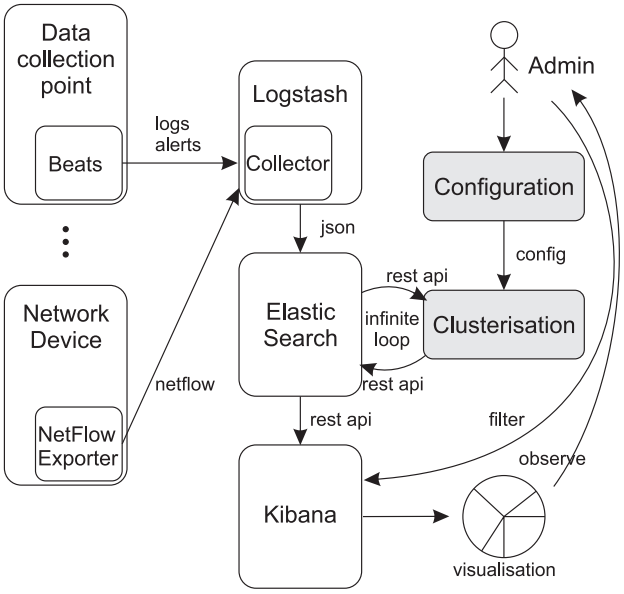

Fig. 1. The proposed architecture for clustering, visualisation and computer-assisted network security analysis

The original Elastic Stack building blocks are shown in white boxes, while the proposed extension is shown in gray (Fig. 1). As already mentioned, Beats sends data such as system logs or alerts out of the systems that generated them, while Logstash receives the data from many sources, transforms it to a JSON format, and stores the data in ElasticSearch (Fig. 1). The ElasticSearch manifests a REST API that Kibana uses for data visualization, allowing the analyst to filter and search through the data (Fig. 1).

The clustering extension from Fig. 1 is implemented in Python and executed on a separate network host. It communicates with ElasticSearch in the same manner as Kibana,

 ISBN 978-86-7466-970-9

forming an infinite loop that: (1) reads the newly arrived data periodically, as illustrated in Fig. 2, (2) clusters the data, and (3) writes back the information about data affiliation to a cluster.

```
1  while True:
2      payload = {"sort":{"@timestamp" : "desc"}, "
           query" : {"match_all":{}}, "size":member}
3      r = requests.post(f'http://10.10.3.224:9200/{
           index_name}/_search', auth=(username,
           password), json = payload)
4      if r.status_code == 200:
5          data = json.loads(r.content)
6          if 'snort' in index_name:
7      ...
```

Fig. 2. The communication between the ElasticSearch and the clustering extension in Python

The sample configuration is given in Fig. 3. The configuration signals the extension which data should be read, defines the ElasticSearch index, as well as a time period for the next execution. Moreover, it holds clustering parameters.

```
1  {
2  "cluster_config":{
3      "number_of_clusters" : 3,
4      "affinity" : "euclidean",
5      "linkage" : "ward",
6      "method" : "ward",
7      "segment" : 50
8  },
9  "db_config":{
10     "username" : "*******",
11     "password" : "*******",
12     "time_wait" : 180,
13     "index_name" : "snort-2023.02",
14     "src_ip_addresses" : ["most_freq"]
15 }}
```

Fig. 3. The configuration

B. Clustering algorithm

We chose agglomerative hierarchical clustering as the clustering algorithm due to its features [4]. Depending on configuration, we read either netflow or snort alerts.

After fetching, the data is gathered in the form of ensembles prior to clustering. An ensemble is a set of ensemble members, where each member represents a set of semantically related data. The clustering is a process of comparing ensemble members, i.e., the sets of semantically related data, and grouping them based on their similarity. The semantic that we use to assign the data to the ensemble member is the common destination IP address and port. This groups alerts or network flows from the same source into one ensemble, where the ensemble members are based on the destination. Unlike [4], where each time window is divided into 30 timesteps, our segment configuration parameter from Fig. 3 further divides the ensemble member into fixed-size time intervals $t_1 = t_2 = \ldots = t_N$, each with its own d_i number of flows or alerts (Fig. 4).

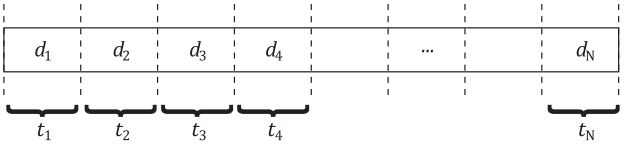

Fig. 4. The ensemble member

An array $D_k = d_i^k, k = 1, 2, \ldots, \mathbf{E}$, which represents a time-domain behavior pattern for a specific network source and its k-th destination, is used to interpret the k-th ensemble member, where $\mathbf{E}$ is a total number of ensembles. All arrays $D_k, k = 1, 2, \ldots, \mathbf{E}$, i.e. ensemble members, are compared, and their mutual distances are calculated using the Dynamic Time Warping (DTW) [4]. DTW forms a matrix $M = [m_{i,j}]$ where each matrix element represents a distance between ensemble member D_i and D_j. Based on the distances, the dendogram is formed. In Fig. 5 an example of a dendogram is shown. A dendrogram illustrates the attribute distances between each pair of consecutively merged classes based on their similarities. To avoid crossing lines, the diagram is graphically organized in such a way that members of each pair of merged classes are neighbors in the x-axis of the diagram.

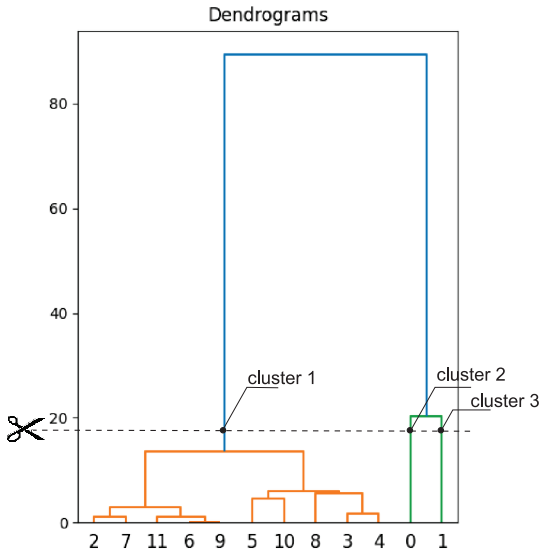

Fig. 5. Example of network traffic dendogram

The proposed system is designed in such a manner to allow setting up two parameters: the number of ensembles to be fetched (line 2 in Fig. 2), and the number of clusters (line 3 in Fig. 3). Due to the complexity of the DTW algorithm, the number of ensembles highly effects the execution time. In the other hand, based on the given number of clusters, our extension will cut a dendogram horizontally to group all ensemble members into a predetermined number of clusters (Fig. 5).

 ISBN 978-86-7466-970-9

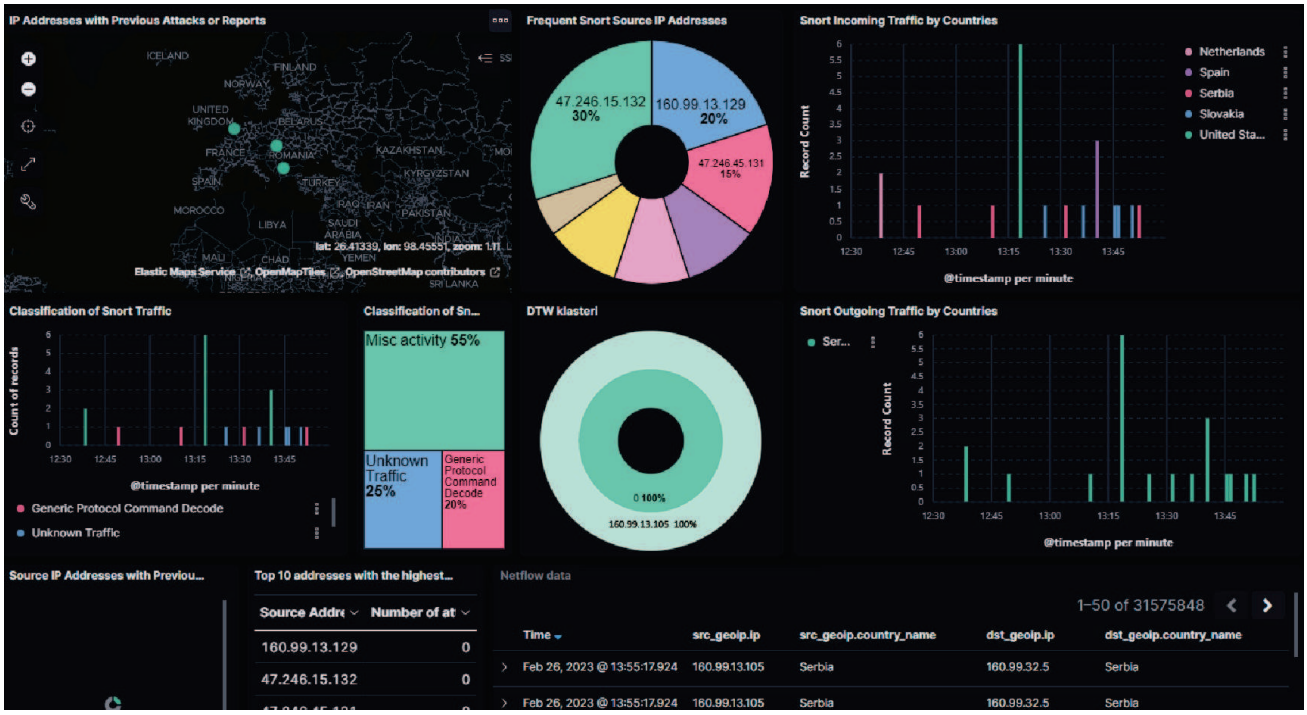

Fig. 6. Data analysis in Kibana by searching and filtering

C. Visualisation and data analysis

As shown in Fig. 1, we selected Kibana for visualization because of its features for additional searching and data filtering. Fig. 6 depicts these capabilities.

Using its intuitive interface, the analyst can select, for example, a particular IP address from the pie chart in Fig. 6 that should be further investigated. The action will activate a data filter, which will update all displayed visual elements. These actions are illustrated as an Admin-Kibana-Visualisation circle in Fig. 1.

As shown in Fig. 6, by selecting a particular IP, the flow records in the bottom-right corner will be filtered to display only relevant records that give the analyst further overall insight into the behavior of one suspicious network node.

By setting up the security analysis environment in the proposed manner, we enabled a low-cost and intuitive visual solution for network traffic and security audits, extended with an advanced clustering algorithm. Clustering in this environment helps the analyst to spot the untypical behavior of network devices that should be further investigated.

IV. Case study

In the case of a port scan attack, we show how the proposed architecture supports identifying the threat. In our case study, we use both Netflow and Snort alerts, gathered at the Faculty of Electronic Engineering, University of Nis, Serbia. The data pipeline is implemented as shown in Fig. 1. The data is collected, clustered, and eventually visualized in the proposed manner. The clustering extension is executed on a PC with Intel(R) i5-2400 CPU @3.10GHz and 32GB of RAM.

The observed network consists of several hundred hosts, where we staged a scene for a port scan attack for the purpose of demonstration. We used a host **A** with an IP address of 160.99.13.105 as an attacker and a host **V** 160.99.32.5 as a victim. The **A** is a Kali Linux host, which executes the well-known port scan *nmap* as ”*nmap* -sV –min-rate 5000 160.99.32.5 -Pn”, while the **V** is a regular network host. An attack with the given parameters lasted for 7 seconds and generated 2,017 flows from host **A** to host **V**, which were recorded in ElasticSearch.

Using a configuration from Fig. 3 we set the Netflow as a data source by defining an index with ”index_name” parameter equal to ”netflow-2023.02”. The extension component is designed to sense the input data format and automatically configure the fields to be used for further analysis, in this case Netflow.

We varied the number of ensemble members by setting the parameter ”members” from the 2nd line in Fig. 2 and measured the execution time. The results are given in Table I. Table I displays the CPU execution time as well as the time required to transfer the data between ElasticSearch and the extension component via REST API (Fig. 1). On the one hand, the table shows that the REST API took up to 30 seconds to complete. This time depends on the current ElasticSearch load. On the other hand, the CPU execution time increases with the increase in the number of ensembles, and it can be seen that for 100 members, it exceeds 2 minutes. As the clustering should be periodically invoked, we used 100 members for further analysis.

Fig. 6 shows the Kibana interface with the pie chart that

 ISBN 978-86-7466-970-9

members	10	20	30	40	50	60	70	80	90	100
CPU time	1.60	5.65	12.10	21.14	32.90	47.51	64.04	81.82	106.51	130.31
REST API	11.30	30.46	9.85	11.24	23.19	29.94	22.36	20.04	23.01	25.92

TABLE I
CLUSTERING EXECUTION TIME FOR VARIABLE NUMBER OF FLOWS IN SECONDS

shows the number of clusters and relevant IP addresses (the pie chart in the middle). It can be noticed that in the case of a port scan attack, there is one cluster that groups the most of IPs. This is due to the fact that the attack generated 2,017 flows in a short period of time, and all the flows had very similar behavior patterns, so they were all classified by DTW in the same class.

The resulting pie chart for "normal" network behavior, where we don't have such a burst of similar network flows, is shown in Fig. 7. From Fig. 7 it can be seen that there are three classes with several different network end points in each of the classes.

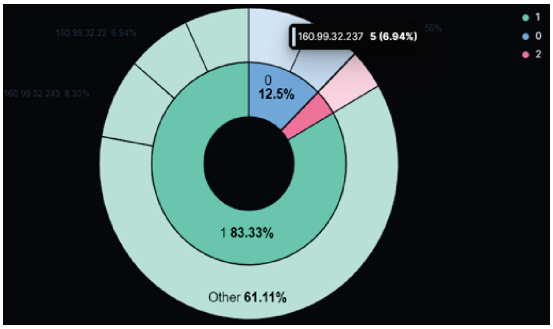

Fig. 7. Network traffic classified in 3 different classes

From the difference in the number of classes detected in the examined time window, it can be seen that the proposed system can be successfully used to detect a burst of similar network flows present in some attack types, i.e., port scanning. The drawback is the long execution time, which implies the small number of ensemble members, so if there is no trigger to signal the potential malicious case, the attack can be missed.

V. CONCLUSION

In this paper, we extended the well-known and well-established data pipeline of open-source Elastic Stack in order to cluster security-related data prior to visualization. The proposed architecture enabled clustering and visualization, followed by data analysis using native Elastic Stack capabilities. The data clustering is implemented using hierarchical clustering. A case study that demonstrates the effectiveness of the proposed solution in the example of a port-scan attack was given. It was shown that the Elastic Stack can be used to successfully identify threats in a cost-effective manner.

ACKNOWLEDGMENT

This work was supported by the Serbian Ministry of Education, Science and Technological Development [grant number 451- 03-68/2022-14/ 200102].

REFERENCES

[1] Cardenas, Alvaro A., Pratyusa K. Manadhata, and Sreeranga P. Rajan. "Big data analytics for security." IEEE Security & Privacy 11.6 (2013): 74-76.
[2] Jing, Xuyang, Zheng Yan, and Witold Pedrycz. "Security data collection and data analytics in the internet: A survey." IEEE Communications Surveys & Tutorials 21.1 (2018): 586-618.
[3] Marija S. Milosevic and Vladimir M. Ciric. "Extreme minority class detection in imbalanced data for network intrusion." Computers & Security 123 (2022): 102940.
[4] Hao, Lihua, Christopher G. Healey, and Steve E. Hutchinson. "Ensemble visualization for cyber situation awareness of network security data." 2015 IEEE Symposium on Visualization for Cyber Security (VizSec). IEEE, 2015.
[5] Crouser, R. Jordan, Erina Fukuda, and Subashini Sridhar. "Retrospective on a decade of research in visualization for cybersecurity." 2017 IEEE International Symposium on Technologies for Homeland Security (HST). IEEE, 2017.
[6] Kwon, Taewoong, et al. "VISNU: A novel visualization methodology of security events optimized for a centralized SOC." 2018 13th Asia Joint Conference on Information Security (AsiaJCIS). IEEE, 2018.
[7] Zhang, Tianye, et al. "A survey of network anomaly visualization." Science China Information Sciences 60 (2017): 1-17.
[8] Zhou, Fangfang, et al. "ENTVis: A visual analytic tool for entropy-based network traffic anomaly detection." IEEE computer graphics and applications 35.6 (2015): 42-50.
[9] Gavrilovic, Nadja, Vladimir Ciric, and Nikola Lozo. "Snort IDS system visualization interface for alert analysis." SJEE 19.1 (2022): 67-78.
[10] Angelini, Marco, Nicolas Prigent, and Giuseppe Santucci. "Percival: proactive and reactive attack and response assessment for cyber incidents using visual analytics." 2015 IEEE Symposium on Visualization for Cyber Security (VizSec). IEEE, 2015.
[11] Theron, Roberto, et al. "Network-wide intrusion detection supported by multivariate analysis and interactive visualization." 2017 IEEE Symposium on Visualization for Cyber Security (VizSec). IEEE, 2017.
[12] Gautam, Bishnu P., and Shiratori Norio. "Suessa: Sustainable & ultra-elastic stack security architecture for securing iot networks of future smart cities." 2020 Eighth International Symposium on Computing and Networking Workshops (CANDARW). IEEE, 2020.
[13] Schüssler, Janik, et al. "Visualizing information for enterprise architecture design decisions using elastic stack." 17th International Conference Perspectives in Business Informatics Research (BIR 2018) Stockholm, Sweden, September 24-26, 2018.
[14] Subramanian, Kiruthiga, and Weizhi Meng. "Threat Hunting Using Elastic Stack: An Evaluation." 2021 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI). IEEE, 2021.

 ISBN 978-86-7466-970-9