

AN ANALYTICAL FRAMEWORK FOR DATA COLLECTION AND ANALYSIS IN IP NETWORK

Matúš ČAVOJSKÝ*, Martin HASIN**, Gabriel BUGÁR*

*Department of Electronics and Multimedia Telecommunications, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Némcovej 32, 042 00 Košice, Slovak Republic, Tel. +421 55 602 2943,
E-mails: matus.cavojsky@tuke.sk, gabriel.bugar@tuke.sk

**Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, Tel. +421 55 602 7648,
E-mail: martin.hasin@tuke.sk

ABSTRACT

The primary focus of the study is to investigate the importance of data collection and analysis in IP networks for detecting, identifying, and responding to potential cyber attacks. It examines the use of the Suricata system's integration with the process of sending detected anomalies to the non-relational Elasticsearch database. The research also looks into the use of Data Lakes, which are centralized storage systems capable of securely storing and analyzing massive amounts of IP traffic data in their native format. An experimental environment is presented, featuring the Elasticsearch database, REDIS cache, and Suricata IDS tool, to conduct experiments. The findings show that combining Suricata with Elasticsearch and Redis cache results a suitable combination, leading to enhanced performance and increased analysis accuracy. In conclusion, by leveraging the strengths of these technologies, it is possible to establish a robust and efficient infrastructure that effectively assists network administrators to safeguard networks against various cyber threats in the network environment.

Keywords: Anomaly Detection, Cybersecurity, Database, Data Lakes, Machine Learning, nDPI

1. INTRODUCTION

As people become increasingly connected to computer networks, the utilization of the internet is growing day by day. This revolution has led to a greater reliance on the global internet to complete personal and professional tasks [1]. As network technology advances rapidly, attackers are developing more sophisticated techniques than defenders. While firewalls and packet filtering based on routers are critical components of a comprehensive network security framework, they are insufficient on their own. As the number of threats targeting computer networks continues to increase, the importance of analyzing and collecting data from IP networks to detect, identify, and respond to potential cyber attacks quickly has become more critical than ever [2]. NFStream plays a vital role in this process by providing efficient and effective network data analysis capabilities [3]. Moreover, implementing robust cybersecurity measures, including intrusion detection and prevention systems (IDS/IPS) such as Suricata and Snort, is critical. With the growing amount of data generated by network devices, it is essential to have a centralized repository to store and analyze IP traffic [4]. Datalakes, a large, centralized data repository that allows for flexible querying and analysis, present a novel approach for this purpose. The use of a datalake benefits the future development of deep neural network (DNN) models by providing easy accessibility to a variety of data sources, including structured, semi-structured, and unstructured data [5]. Datalakes enable researchers to explore and experiment with different datasets, leading to more comprehensive and accurate training of DNN models. The scalability of datalakes allows for the storage of massive datasets required for training complex models [6].

2. CYBERSECURITY METHODS

Cybersecurity systems often utilize advanced analytics and machine learning techniques to identify and mitigate security threats in real-time. Both datalakes and cybersecurity systems require scalable and flexible architectures that can handle large volumes of data and support complex analytics workflows [7]. By analyzing hidden trends, similarities, and other insights, these systems provide companies with the ability to optimize their data and uncover new zero-day attacks.

Two distinct approaches exist for protecting computer systems and networks from potential cyber threats - proactive and reactive [8]. In reactive cybersecurity, the focus is on responding (reacting) to a cyber attack after it has occurred. This approach includes measures such as incident response plans, data backup and disaster recovery procedures to minimize the impact of an attack. Reactive cybersecurity also includes responding to the root cause of the attack and implementing measures to prevent similar attacks from happening again in the future. A typical example of a reactive cybersecurity system is an intrusion detection system (IDS). IDS is a security system that monitors network traffic, system events, and other activities on a network to identify and alert security personnel about potential security breaches, such as unauthorized access, network scanning, and malware infections after an intrusion has occurred or an attack has been launched [9].

Proactive cybersecurity involves the implementation of various security measures such as firewalls, intrusion detection systems and anti-virus software. An intrusion prevention system (IPS) is a security system that, in contrast with an IDS, not only detects but also takes proactive

measures to prevent potential security breaches by analyzing network traffic and blocking or dropping suspicious or malicious traffic in real-time to prevent attacks from happening [10]. In addition, proactive cybersecurity involves conducting regular security assessments, identifying vulnerabilities and fixing them before attackers exploit them. Due to the common occurrence and relative ease of cyber attacks, particularly on insecure operating systems like Windows, computer and network security require an assortment of specialized tools designed to prevent and detect intrusions [11].

Suricata is an open-source intrusion detection and prevention system (IDPS) that uses rule-based techniques to detect and prevent security threats [12]. It is designed to monitor network traffic in real-time and analyze it for known attack patterns and vulnerabilities. As an IDPS, Suricata can be used to detect, alert administrators, and take proactive measures to potential security breaches by analyzing network traffic and comparing it against a set of predefined rules in real-time. It is proficient at identifying potential threats within network traffic and generating alerts or logs for further investigation. However, the information it provides is often in raw log or alert format, which can be challenging to interpret, especially when dealing with a large volume of data.

In recent years, the amount of data generated by businesses and organizations has increased exponentially, leading to a demand for more efficient methods of data storage and analysis. One such solution is a Data Lake, a large-scale storage system that allows storing vast amounts of data in their native format without the need for prior structuring or processing [13]. A data lake serves as a repository for all types of data, including structured, unstructured, and semi-structured data, providing a flexible and cost-effective way to store and manage data. Data lakes are used in a variety of industries, including finance, healthcare, retail, and technology.

Big data lakes possess a unique feature of facilitating machine learning tools at the frontier layers, with the ultimate aim of aiding the extraction of valuable insights from the core big data repositories [14]. These insights can be leveraged for knowledge discovery and effective decision-making processes [15]. Both NFStream and datalakes enable the collection and processing of large volumes of data, allowing for in-depth analysis, insights and possibly a machine learning based cybersecurity solutions. Elasticsearch provides a powerful and scalable platform for indexing and storing Suricata logs efficiently. It is optimized for fast searches, real-time analysis, as well as the search and query capabilities of Elastic Stack which enable analysts to correlate Suricata alerts with other security-related data sources [16]. Historical searches can be utilized to identify recurring threats or unusual patterns of behavior. A broader perspective helps to recognize complex attack patterns and understand the full scope of an incident.

The goal of this paper is to demonstrate how Suricata along with Elasticsearch and Redis cache is the most effective combination, significantly improving performance and reduce the packet loss when compared to more traditional solutions like SSD log cache or RAM disk.

3. EXPERIMENTAL ENVIRONMENT

When creating an experimental environment on which experiments will be carried out with the Elasticsearch database, the REDIS cache and the IDS tool Suricata, it is necessary to follow the following steps:

- Hardware and software requirements: This option includes the necessary servers or virtual machines with sufficient CPU capacity, memory and storage space, as well as the necessary operating system and network configurations.
- Suricata IDS Installation: Install Suricata on the designated server using the operating system's package manager.
- Installing Elasticsearch: Installing Elasticsearch on a separate server.
- Installing the Redis cache: Installing the Redis cache on a separate server with enough RAM.
- Configuration of installed modules, setting of required IP addresses, and connectors used for communication between installed applications.

The experimental environment for conducting the experiment consists of an HP DL580 G8 server with 328GB RAM and 4x Intel(R) Xeon(R) CPU E5-4610 v0 @ 2.40GHz and the Ubuntu 20.04 operating system with Linux Kernel 5.15. The server includes an Intel Corporation 82599ES network card, which enables connection of 8 processor cores for processing network packets. The RSS technology ensuring processing on multiple IDS threads by the Suricata 6.0.8 system enables network traffic to be analyzed on 48 cores.



Fig. 1 Design of experimental infrastructure

Figure 1 illustrates the infrastructure under consideration, demonstrating the transmission of network traffic to the Suricata server through Cisco SPAN technology. Subsequently, the server processes this network traffic and the measurement results are stored in a non-relational database. A cluster of 3 Elasticsearch servers is considered as a non-relational database, where data are stored and indexed. Data are stored in a separate index with a size of 30 shards.

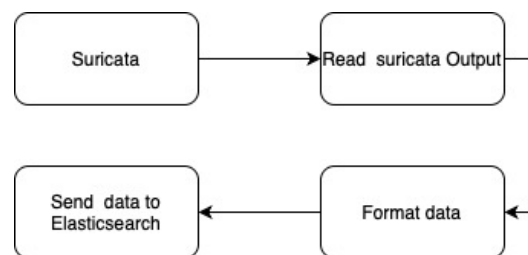


Fig. 2 Design of Suricata infrastructure

Figure 2 shows the considered infrastructure, which collects data from the Suricata system, then processes them

and sends them to the environment of the non-relational elasticsearch database. Two approaches were considered for this kind of data collection:

- Measurement 1 - Data collection using json parsing of Suricata's eve.log file, writing files to SSD
- Measurement 2 - Collecting data by parsing json eve.log of Suricata system with writing files to RAM
- Measurement 3 - use of the REDIS database, which serves as a RAM-oriented database for fast data access

To assess and evaluate the testing environment's quality, we selected samples of network traffic measured in real-world operations, where the threat count was known. These samples, presented in Table 1, comprised two PCAP files, which were subsequently duplicated to the network adapter and analyzed using the Suricata system.

Table 1 Samples used

Sample size:	The number of known threats:
401 GB	191185868
400 GB	191083481

The PCAP files with pre-recorded threats were converted to the network adapter using tcpdump and transferred to the network probe's network adapter using CISCO SPAN technology. The used Cisco C4900 switch and the connection between the Suricata server and the server simulating network traffic consisted of 10Gb/s Ethernet formed using optical converters. The tcpdump application is used to analyze and reproduce these threats in the network. This assured the same parameters for every scenario. These threats can also be replicated in several cycles, which makes it possible to monitor how the proposed system can detect a specific type of threat. Replicating a sample multiple times, especially in several cycles, allows for a more comprehensive evaluation of the proposed system's robustness and effectiveness in consistently detecting a specific type of threat, ensuring reproducibility. Setting up tcpdump and sending data to the network adapter was done using the command:

```
tcpdump -i eth7 -tK live-network.pcap
```

3.1. Experiment 1

The first measurement consisted in creating an experimental environment consisting of a Suricata server, which recorded the measurement result on a local SSD disk on the given server. The recorded data were stored in the eve.log file. Subsequently, using the logstash application, these data were read and sent to the non-relational Elasticsearch database.

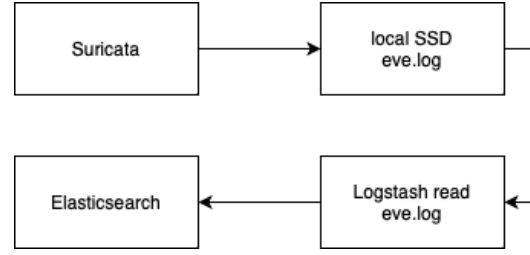


Fig. 3 Design of infrastructure Experiment 1

The wiring of components used for data loading and sending them to a non-relational database is depicted in Figure 3. Setting up the Suricata system consisted in setting up a module ensuring JSON export to a file. The eve-log module was chosen as the export, which allowed us to convert the recorded samples into JSON format and then save them to a file.

```

outputs:
- eve-log:
    enabled: yes
    filetype: regular
    filename: eve.json
  
```

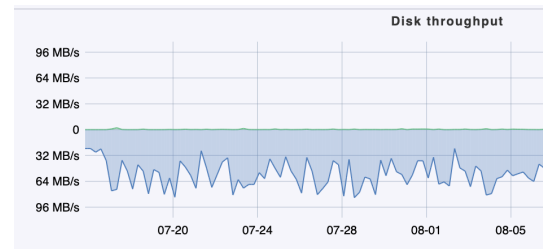


Fig. 4 Disk utilization

Figure 4 shows the utilization of the disk on which the logs were being written. The average value for data writing was 34MB/s, where in short-term peaks the process consumed up to 140MB/s of writing to the disk. Subsequently, this data were loaded using logstash and then written to a non-relational database.

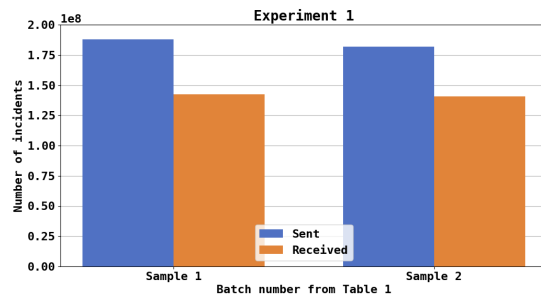


Fig. 5 Result of Experiment 1

The number of threats captured and saved by the Suricata system is illustrated in Figure 5. Using this configuration and saving data to disk storage, about 74 percent of all threats in the measured samples were captured. The sent and received packets show us the packet loss due to inefficient bandwidth or large delays in the pipeline.

3.2. Experiment 2

The second experiment consisted in creating a measurement similar to the first experiment where the Suricata system was used, which generated outputs to the eve.log file. However, the setting of the measuring system was changed so that the LOG file is written to the learned RAM disk. A 100 GB RAM disk was stored in the RAM memory, enabling rapid simultaneous data access for both reading and writing.

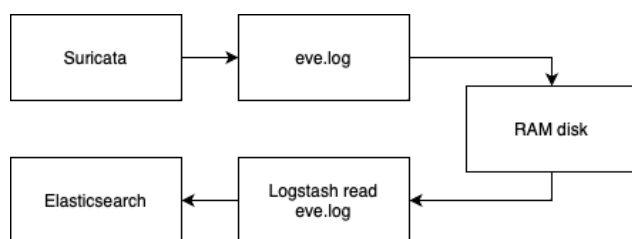


Fig. 6 Design of infrastructure Experiment 2

The Figure 6 shows a block diagram of how data is stored from the Suricata system and loaded by the logstash system. The number of threats captured and saved by the Suricata system in the second experiment is depicted in Figure 7. The sent and received packets indicate any packet loss caused by insufficient bandwidth or significant pipeline delays. When using this configuration and saving data to the RAM disk, threats were captured in the range of 89 to 90 percent of all known threats entering the measurement.



Fig. 7 Result of Experiment 2

3.3. Experiment 3

The last experiment consisted in the inclusion of a REDIS server, which served as an intermediate link during data transfer. The advantage of the REDIS server is that it is fully oriented to work with RAM, where data from this database are primarily stored in RAM memory and subsequently copied to disk storage at regular intervals. The advantage of this database is also the method of indexing, which supports writing and reading data at high speed. On the Suricata system side, a module was used that allows the obtained data to be transformed into the REDIS database language and subsequently to save these outputs in this database.

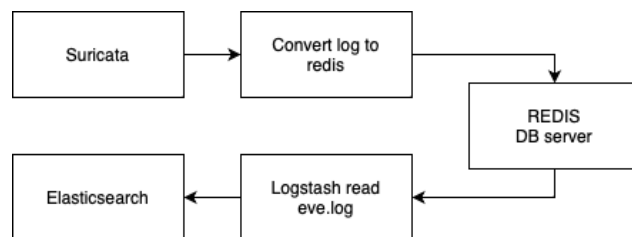


Fig. 8 Design of infrastructure Experiment 3

Figure 8 shows the data flow diagram between the Suricata system and a non-relational database. Using the module for REDIS, the Suricata system allows you to send the results of the analysis to the memory-oriented REDIS database, where it is necessary to set:

```

- eve-log:
  enabled: yes
  filetype: redis
  threaded: true
  redis:
    server: 127.0.0.1
    port: 6379
    key: suricata
  
```

Since the memory-oriented REDIS database enables the automatic storage of results from the Suricata system, it is necessary to periodically delete the data stored in the database. The deletion of this data was realized by creating a module in the python language, which ensured the deletion of records older than 15 minutes.

```

import redis
r = redis.StrictRedis(host='localhost',
port=6379, db=0)
r.flushdb()
  
```

The process of deleting old records will prevent the continuous increase in the capacity of the memory-oriented REDIS database, which subsequently stores data from RAM memory to local HDD storage.

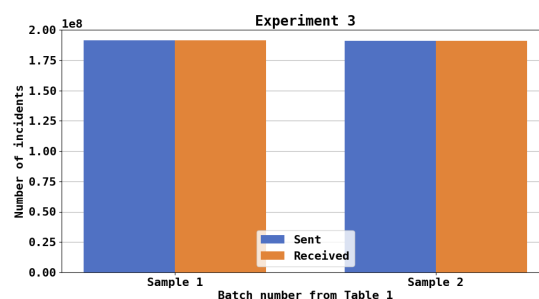


Fig. 9 Result of Experiment 3

Figure 9 illustrates the number of threats captured and saved by the Suricata system in the third experiment. Using this configuration captured 99.99 percent of all known threats, the lowest loss from any previous experiments. The REDIS database was significantly faster and more applicable to delay sensitive application than RAM disk or an SSD log file.

4. RESULTS

The aim of this work was to experimentally connect and verify the functionality of the Suricata system and the sending of obtained anomalies to the non-relational Elasticsearch database. When the system was connected, three experimental situations were created as to how it is possible to obtain measured data from the Suricata System and then further process and send it to another data processing module. An experiment focused on the use of the REDIS memory-oriented database came out as the most advantageous version of data storage and retrieval. Using this experiment, it was possible to prove the accuracy of over 99.99 percent of the detected anomalies compared to the reference model.

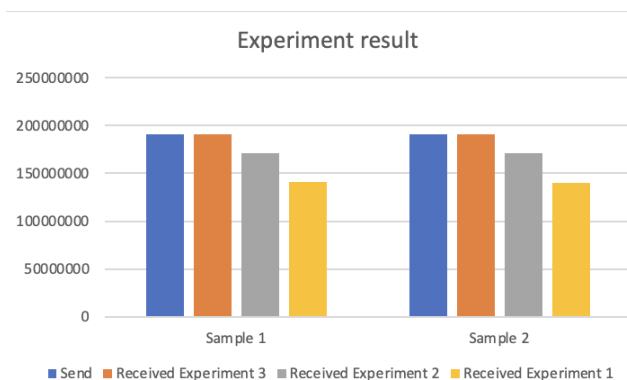


Fig. 10 Result of all Experiments

Figure 10 is a graph of the success of recording and saving anomalies from the measured data sample. The least accurate were the measurements that save the loaded data to the eve.log file in JSON format. This process came in as the slowest even when using fast memory storage. The disadvantage of storing recorded anomalies in JSON format is the sub-optimal transformation of samples into JSON format. The process that deals with data transformation can only work on one core, which caused high data loss at higher data flow. Data reduction was also possible by turning on threading, which made it possible to write transformed JSON anomalies in threads to different files. However, this approach could not eliminate the use of the JSON module, which transforms data on only one core of the operating system. The use of threading also slowed down the system by creating a lot of IOPS that the operating system kernel needed to write to the internal disk. The best form was the use of a memory-oriented REDIS database, which served as an intermediate link in which the Suricata system writes data and the logstash system reads the obtained data and applies the required filters to this data. By using the memory-oriented REDIS database, it is possible to easily optimize the deletion of outdated data that the system was already able to analyze and store in a non-relational database. By using this model, it is not necessary to restart the Suricata system when deleting outdated data, which makes it possible to analyze network traffic without downtime or loss of information about the given network.

5. CONCLUSION

In conclusion, implementing Suricata with Elasticsearch and Redis cache is the most suitable combination that can significantly increase performance and increase the accuracy of analysis. By leveraging Suricata's advanced intrusion detection capabilities combined with Elasticsearch's fast indexing and querying capabilities, security mechanisms quickly identify potential threats and respond in real time. Additionally, partially caching Redis into the infrastructure can further improve its performance by reducing the load on Elasticsearch while maintaining fast access to critical data. This cache layer can be especially useful in large deployments where performance and scalability are paramount. Suricata's overall implementation with Elasticsearch and Redis cache provides a robust and efficient solution for network security monitoring and analysis. Leveraging the strengths of each technology, it is possible to create a highly secure infrastructure that is fast and reliable, ultimately increasing the effectiveness of the ability to protect your networks from a wide range of cyber threats in the network environment. The integration of Suricata with Elasticsearch offers the potential to significantly enhance cybersecurity by leveraging the rich data generated by Suricata for training and enhancing IPS machine learning models. This synergy enables the development of more adaptive and predictive threat detection systems, better equipped to identify emerging threats and defend against sophisticated cyberattacks.

ACKNOWLEDGEMENT

This publication has been published with the support of the Operational Program Integrated Infrastructure within project: Research in the SANET Network and Possibilities of Its Further Use and Development (ITMS code: 313011W988), co-financed by the ERDF. Development and innovation of TUKE simulation environment in the field of technical sciences (004TUKE-2-1 / 2021), and the Slovak Academy of Sciences under Grant VEGA 1/0685/23.

REFERENCES

- [1] KHANNA, P.: *Connectography: mapping the global network revolution*, Weidenfeld & Nicolson, No. 2016 (466) ISBN 978-1474604253 <https://doi.org/10.1111/1468-2346.12768>
- [2] JAAFAR, G. A., et al.: *Review of Recent Detection Methods for HTTP DDoS Attack*, Journal of Computer Networks and Communications, grade. 2019 **doi.org**, No. 18 (1) 20191–10 <https://doi.org/10.1155/2019/1283472>
- [3] ZIED, A. – PEKÁR, A.: *NFStream*, Computer Networks, grade. 204, No. 18 (2) 2022108719 <https://doi.org/10.1016/j.comnet.2021.108719>
- [4] von SOLMS, R. – van NIEKERK, J.: *From Information Security to Cyber Security*, Computers & Security, grade. 38, No. 18 (10) 201397–102 <https://doi.org/10.1016/j.cose.2013.04.004>
- [5] KILINCER, I. F., et al.: *Machine Learning Methods*

- for Cyber Security Intrusion Detection: Datasets and Comparative Study, Computer Networks, grade. 188 , No. 18 (4) 2021107840 <https://doi.org/10.1016/j.comnet.2021.107840>
- [6] VAJDA, D., et al.: *Towards Machine Learning-based Anomaly Detection on Time-Series Data*, Infocommunications journal, grade. 13 , No. 18 (1) 202135–44 <https://doi.org/10.36244/ICJ.2021.1.5>
- [7] PEKÁR, A., et al.: *Adaptive Aggregation of Flow Records*, Computing and Informatics, grade. 37 , No. 1 (1) 2018142–164 https://doi.org/10.4149/cai_2018_1_142
- [8] KHUMMANEE, S., et al.: *Towards a new design of firewall: Anomaly elimination and fast verifying of firewall rules*, The 2013 10th International Joint Conference on Computer Science and Software Engineering (JCSSE), Khon Kaen, Thailand **2013**, No. 93-98 () doi:10.1109/JCSSE.2013.6567326
- [9] PAPADAKI, M. – FURNELL, S.: *IDS or IPS: What Is Best?*, Network Security, grade. 2004 doi.org, No. 7 (7) 200415–19 [https://doi.org/10.1016/S1353-4858\(04\)00106-0](https://doi.org/10.1016/S1353-4858(04)00106-0)
- [10] ASHOOR, A. S. – GORE, S.: *Difference between Intrusion Detection System (IDS) and Intrusion Prevention System (IPS)*, In: Wyld, D.C., Wozniak, M., Chaki, N., Meghanathan, N., Nagamalai, D. (eds) *Advances in Network Security and Applications CNSA 2011*, No. Communications in Computer and Information Science (Springer, Berlin, Heidelberg) 2011196 https://doi.org/10.1007/978-3-642-22540-6_48
- [11] CANETTI, R., et al.: *Proactive security: Long-term protection against break-ins*, RSA Laboratories' CryptoBytes , No. 3 (1) 19971-8 https://www.researchgate.net/publication/2708104_Proactive_Security_Long-term_protection_against_break-ins
- [12] GALINEC, D., et al.: *Cybersecurity and Cyber Defence: National Level Strategic Approach*, Automatika, grade. 58, č. 3 (Crossref) doi.org, No. 18 (7) 2017273–286 <https://doi.org/10.1080/00051144.2017.1407022>
- [13] MILOSLAVSKAYA, N. – TOLSTOY, A.: *Big Data, Fast Data and Data Lake Concepts*, Procedia Computer Science **88**, No. 2016 (300-305) ISSN 1877-0509 <https://doi.org/10.1016/j.procs.2016.07.439>
- [14] MATHIS, Ch.: *Data Lakes*, Datenbank-Spektrum **Jeju Island**, No. Korea (South) (289) 20211610-1995 <https://doi.org/10.1007/s13222-017-0272-7>
- [15] CUZZOCREA, A.: *Big Data Lakes: Models, Frameworks, and Techniques*, 2021 IEEE International Conference on Big Data and Smart Computing (BigComp) **Jeju Island**, No. Korea (South) () 20211-4 <https://doi.org/10.1109/BigComp51126.2021.00010>
- [16] KONONENKO, O., et al.: *Mining modern repositories with elasticsearch.*, Proceedings of the 11th working conference on mining software repositories. , No. () 2014328–331 <https://doi.org/10.1145/2597073.2597091>

Received June 05, 2023, accepted November 27, 2023

BIOGRAPHIES

Matúš Čavojský received M.Sc. degree in computer networks from Technical University of Košice, Slovakia in 2022. Currently, he is a PhD student at the Department of Electronics and Multimedia Telecommunications of the Faculty of Electrical Engineering and Informatics, Technical University of Košice since 2022. His scientific research is focused on data analysis and artificial intelligence for cybersecurity. In addition, he also investigates questions related to split and edge computing systems.

Martin Hasin is a PhD student at the Department of Computers and Informatics at Faculty of Electrical Engineering and Informatics, Technical University of Košice since 2020. His research interests data analysis and artificial intelligence for cybersecurity.

Gabriel Bugár received M.Sc. and Ph.D. degrees in electronics from Technical University of Košice, Slovakia in 2007 and 2010, respectively. Currently, he is Research Assistant at the Department of Electronics and Multimedia Telecommunications, Technical University of Košice.