Network Security Traffic Analysis Platform - Design and Validation

Zineb Maasaoui*,Anfal Hathah*†, Hasnae Bilil*, Van Sy Mai*, Abdella Battou*, Ahmed lbath‡
Advanced Network Technologies Division, National Institute of Standards and Technology, MD, USA
†LIG/MRIM, CNRS, Grenoble Alpes University, Grenoble, FRANCE
‡Department of Computer Engineering and Sciences, Florida Institute of Technology, FL, USA
Email: {zineb.maasaoui, anfal.hathah, hasnae.bilil, vansy.mai, abdella.battou}@nist.gov,
ahmed.lbath@univ-grenoble-alpes.fr

***Abstract*—Real-time traffic management and control have become necessary in today's networks due to their complexity and cybersecurity risks. With the increase in Internet use, threats are more prevalent and require real-time detection and analysis to prevent network intrusions. As the number of data flow increases, the number and the types of attacks increase, which makes detecting intrusions challenging. Therefore, over the last years, many researchers have focused on different ways to detect and more importantly prevent these intrusions**

In this work, we describe the design and evaluation of a network security traffic analysis platform (NSTAP) that collects, searches, and analyzes traffic data in real time in order to filter out malicious flows. Through charts, tables, histograms, and other visualization methods, we demonstrate that the platform can produce powerful and useful insights with simple time-domain analytics of large data volumes. This work is intended to be the foundation for more automation tools based on machine learning.

***Index Terms*—Network, Security, Intrusion Detection, Traffic management, Machine learning**

I. Introduction

Threat detection is a fundamental component of the current network security ecosystem. Networks are becoming more vulnerable because of the steady increase in attack attempts. The need for a real-time solution for detecting, managing, and controlling the traffic in a network has become necessary. Attacks are growing in sophistication, leading to the need to understand and identify the various properties of the network's flows.

Several studies have been published on intrusion detection systems (IDS). The used techniques are generally signature-based and can only detect known threats. In the early research, the author in [1] based his study on predefined rules to represent the characteristics of abnormal data. The matching of these rules with the data profile allows to decide whether there is an intrusion or not. Another approach [2] aims at improving the detection by combining signature-based methods and immune-based intrusion detection methods to improve the detection rate and reduce the false alarm rates. In addition, machine learning-based approaches are used to detect zero-day intrusions in the network traffic [3]. While in [4] the authors studied different supervised algorithms applied to KDD99 dataset and showed no high detection rate for each class of this dataset. The authors in [3], [5] presented a comparative study of different machine learning techniques for anomaly detection.

All these strategies require a large dataset of network flows. There are a few available datasets, some of them are outdated, others lack traffic diversity, and the more recent ones are still being evaluated.

In this work, we aim to develop a network security traffic analysis platform (NSTAP) that will be deployed inside or at the edge of a network to collect and analyze real-time network traffic flows. It will also archive the data for batch processing in a HADOOP environment. NSTAP is able to combine its collected data with existing datasets to provide a testbed for real-time network traffic flow analysis.

II. Related Work

With the rise of Internet usage, network attacks have become more frequent, and Distributed Denial of Service (DDoS) attacks make up a big fraction of these attacks. Consequently, multiple studies have been conducted to model attacks and enhance networks security [6]. For instance, the authors in [7] proposed a lightweight algorithm to protect communication networks against DDos attacks.

A part of the current IDS studies focused on traffic monitoring by offering a traffic visualization and analysis [8]. Other studies focused on new ways to collect, search, and visualize the streaming network traffic – Elastic search and Kibana for example. The authors in [9] for example, are able to process streaming data and provide intuitive visualizations to facilitate the detection of attacks in real-time. However, these methods of visualization remain inefficient (not scalable) when the amount of data is large. Also, they did not provide access to the traffic history. While the authors in [10] treated the traffic as a collection of flows and looked for significant changes in the traffic patterns. They used the data to derive a model of normal behavior based on past traffic history. Then, they looked for significant changes in short-term behavior that are consistent with that model. This approach can detect accurately significant changes in traffic, albeit with a lot of false alarms in massive data streams. Another study analyses the traffic characteristics based on HADDOP [11]. A traffic monitoring system was developed to collect real-time network traffic using

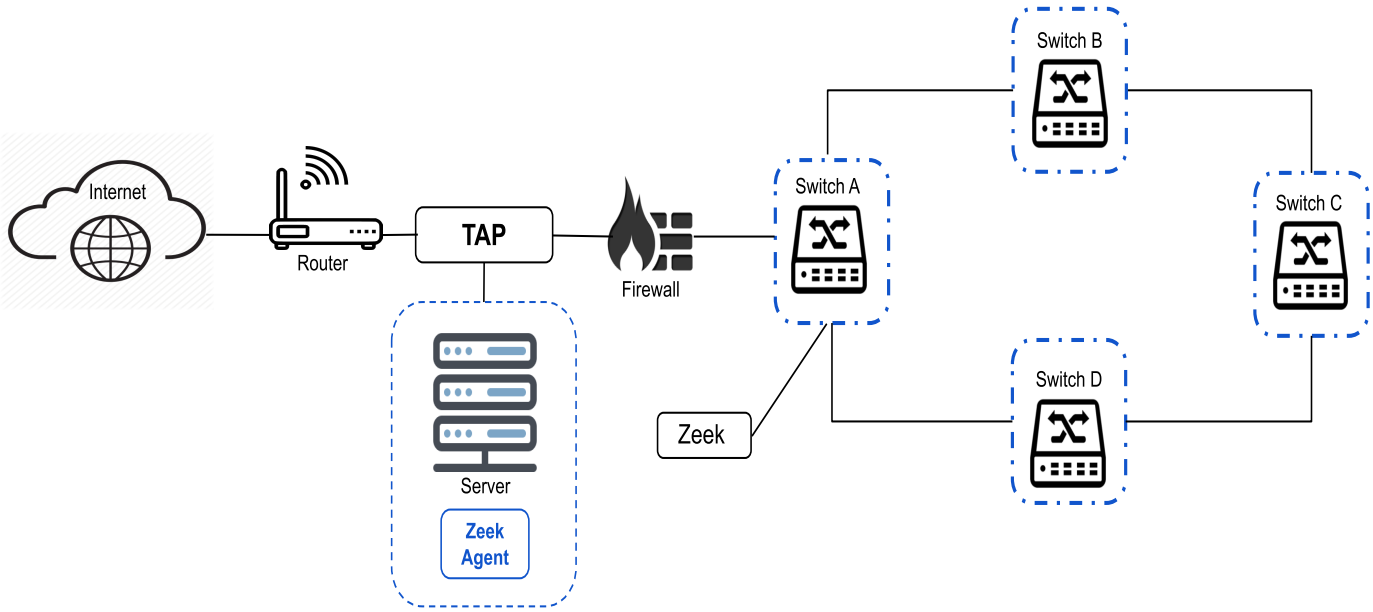

Fig. 1. Network architecture.

SARIMA model to examine the traffic characteristics and user behavior from the perspective of flowsMore.

There are several security platforms on the market – firewalls, intrusion detection systems, virus scanners, etc. There are also numerous open-source tools to collect network data. What we noticed is a lack of tools for real-time analysis of large volumes of network flows.

III. Network Security Traffic Analysis Platform Design

The NSTAP design has two main objectives – inform us about two aspects of our network: (1) Performance (2) Security.

In terms of performance, we would like to monitor the network, get alerts when certain links are about to experience congestion, and receive alerts when certain network elements are starting to drop packets on certain ports. We would like to also have the capability to detect loops and parts of the network that are unreachable. In terms of security, we would like to detect threats and provide incident response for fast remediation.

The performance and security evaluation requires the monitoring of the network at various points and collecting massive volumes of data. This in turn defines our first design requirement of NSTAP we call the data strategy (NSTAP Pipeline) requirement – the ability to ingest large amounts of data. The second design requirement of NSTAP is the processing requirement – the ability to process the data in real time. These two requirements need to scale with network and data collection size.

The performance goal includes a network connectivity analysis to determine the status of all network activities in order to get a picture of bandwidth and resource utilization on the network.

While the security goal is to have a capability that would ingest all data from across the network, normalize it to make it searchable, analyze it for anomalies, and then investigate events and remediate incidents to neutralize attackers. In this paper, we will only discuss network security and leave the performance part of NSTAP for a future publication. In the following, we will provide a description of the NSTAP components and architecture.

A. Data Collection

The data collection module is responsible for collecting network and host data. It is composed of two agents – Zeek and Osquery agents. Both platforms are well-documented in [12] and [13]. We will only describe how we integrated them into the NSTAP.

B. ZEEK Agents

Zeek (previously named Bro) is an open source network analysis framework developed by Vern Edward Paxson a Professor of Computer Science at the University of California, Berkeley. It is released under the BSD license. To collect network data, we tap the network before the firewall using a bidirectional optical tap that we connect to a Linux server where we run a zeek agent as shown in Fig. 1. The reason for this is to get all the attacks including the ones filtered by the firewall. We also deploy what we call a managed zeek probe in every segment in the network using port mirroring. This is a low-cost Raspberry Pi computer running the zeek agent. All of these zeek data collectors/Probes are managed traffic monitoring resources controlled from the NSTAP management system.

C. Osquery Agents

Osquery is a high-performance and low-footprint distributed host monitoring daemon (Osqueryd). Its strength comes from the way it exposes the host operating system as an SQL Database. The benefit of an SQL database API is that we can query our entire host's infrastructure. The daemon aggregates the query results over time and generates logs that indicate state changes in our infrastructure. This allows us to build an intelligent layer on top of Osquery to maintain insight into the security and performance of the entire infrastructure. Notice that through Osquery we have access to the application traffic.

D. Network Security Traffic Analysis Platform architecture

The NSTAP design includes 4 components – (1) Intelligent Agents/Probes for data collection, (2) Pipeline for data strategy, (3) backend for real-time data processing, and (4) a Graphical User Interface (GUI) for representation and access. We had several design choices on how to organize the interactions between these components. The selection of intelligent probes like Zeek and Osquery made our design choice very flexible. Scalability was a major design parameter. We wanted to be able to ingest massive amounts of data. After several evaluations, we used a combination of two approaches:

(1) REST APIs Model

(2) Publish/Subscribe Model

Both approaches have been used in large production environments (large financial applications in banks for example) and have been shown to scale very well. Our choices were also guided by existing open source projects we could use for our implementation. We started with the first model and used the OpenAPI Specification (OAS) to define language-agnostic interface to RESTful APIs. We used it to generate code for the implementation of the green (1), gold (2), and red (3) interactions shown in Fig. 2. We also use the green connections (1) to submit SQL queries for real-time inquiries from the GUI. The collected data was stored in a Mongo Database and accessed through the GUI to the initial evaluation (Fig. 3). We put those queries on schedule to generate both network and host data to start the evaluation and NetOps of our intelligent layer. This first implementation is shown in Fig. 4 in red.

Fig. 2 below shows the real-time section of the NSTAP architecture without the details of its pipeline. The green lines labeled (1) refer to management bidirectional connections using REST APIs. Endpoints exist on the agents and on the NSTAP Backend. These REST API connections allow us to carry configuration changes/updates on the agents and to also receives alerts from the agents such as health status. Both zeek and osquery require some configurations and this approach allows to dynamically make changes and do a restart when necessary. The gold labeled (2) and red labeled (3) connections use a push model to move the data from zeek and osquery respectively to the NSTAP pipeline. We use the green connections labeled (1) to update the data collection push schedule.

We also use the green connections (1) to submit SQL queries for real-time inquiries from the GUI.

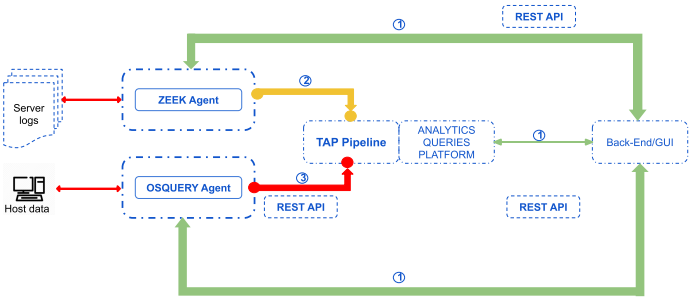

Fig. 2. NSTAP Architecture - Real-time section without the data pipeline.

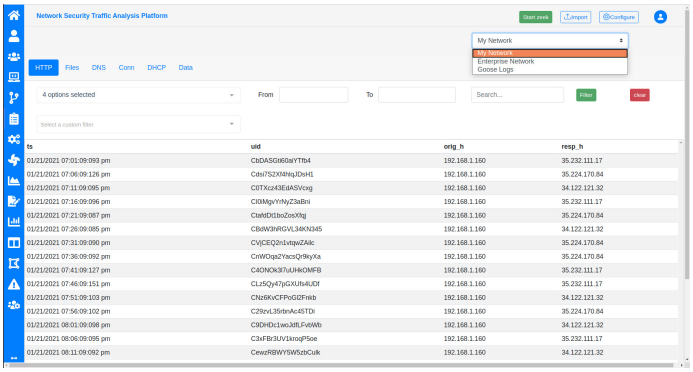

Fig. 3. Screenshot of NSTAP GUI of traffic logs

E. Network Security Traffic Analysis Platform Data Pipeline

The data pipeline includes a load balancer (LB), a Webserver (NGINX) , and a publish/subscribe messaging component (Kafka). These components are selected because of their scalability and have been all tested in various environments. This data pipeline will feed both the real-time section through Fig. 3 and Fig. 4 a database and the batch processing section through a Hadoop environment as shown in Fig. 4.

The batch processing section will be used for unsupervised learning algorithms to augment our automated detection algorithms.

IV. Use case for NSTAP validation - industrial control system

Due to covid-19 crisis, we weren't able to collect data from our Smart Grid laboratory. To test the NSTAP we are using the industrial network shown below in Fig. 5.

The substation's energy network part is shown in Fig. 6. It represents the one-line diagram of a power system used

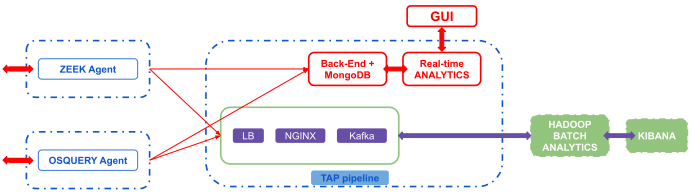

Fig. 4. NSTAP Architecture - With data pipeline and batch processing section.

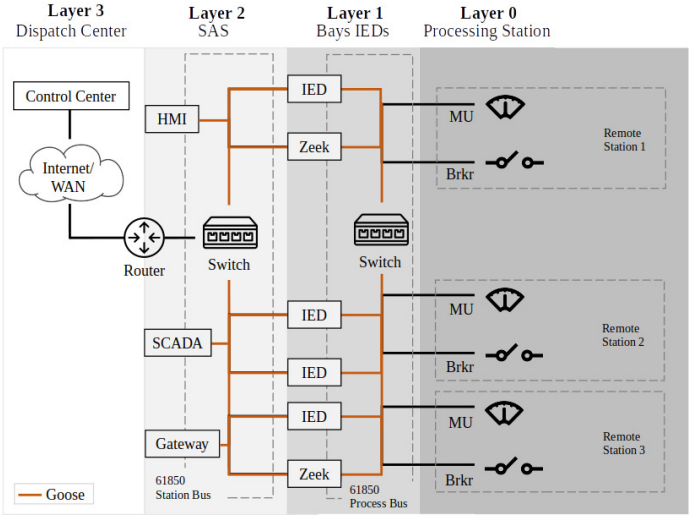

Fig. 5. Industrial control network environment [14].

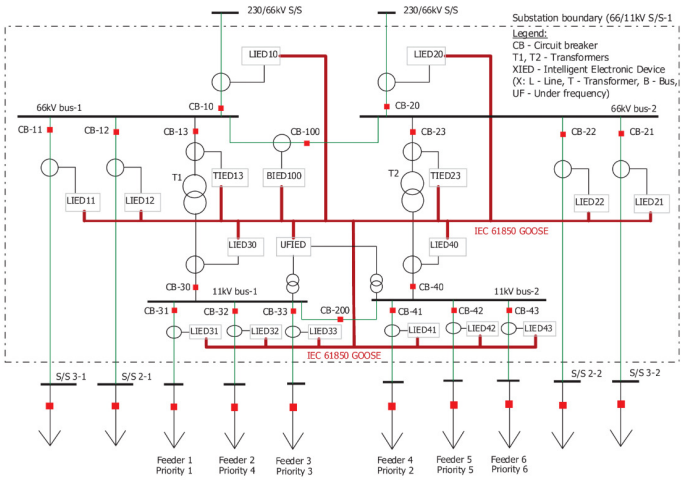

Fig. 6. Substation one-line diagram [15].

for generating network traces. It consists of buses and 18 Intelligent electronic devices (IEDs), each with a unique Mac address. The IEDs communicate among themselves using IEC61850 Generic Object Oriented Substation Event (GOOSE) communication protocol to facilitate information exchange between devices in electrical substations. The generated GOOSE trace file can be analyzed in Zeek to generate goose.log file. These logs consist of a subset of the GOOSE data fields relevant to cybersecurity analysis.

We are feeding the NSTAP with the IEC-61850 dataset [15] using the following simulated scenarios : attack-free, attack-free with disturbance and attack trace with message suppression scenarios. Then, we use NSTAP to analyze the traffic by : (1) Searching the logs from NSTAP logs section (Fig. 10) and (2) Generating graphs.

For the normal scenario, since all the IEDs are running normally, the trace generator program is set to 1 GOOSE frame/second and there are no event changes in the GOOSE

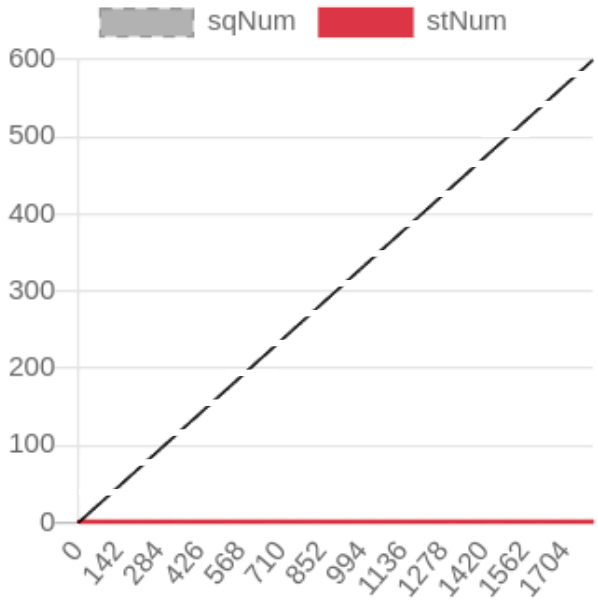

Fig. 7. Screenshot of attack-free Network trace.

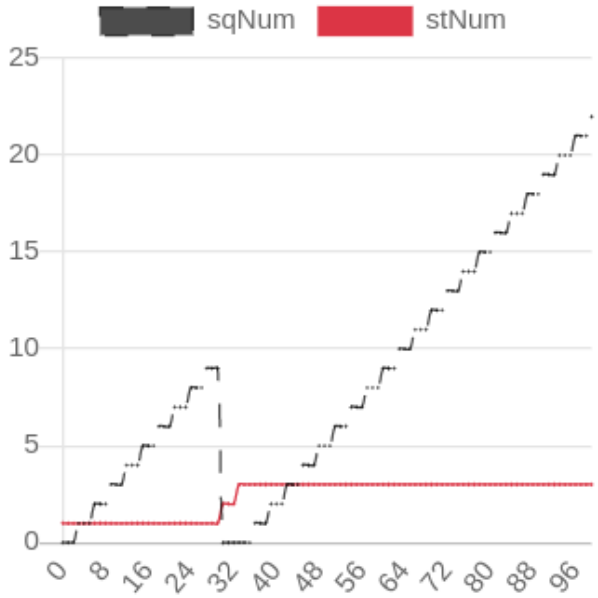

Fig. 8. Screenshot of disturbance Network trace.

dataset. In Fig. 7, SqNum and StNum are correlated with each GOOSE frame and the number of events respectively. With stNum and sqNum are counters that indicate a change within the Goose Frame and the dataset respectively [15]. As for the breaker failure scenario, which is a type of disturbance, a line fault is simulated to activate the breaker failure protection schema. LIED11, represented in Fig.6, sends a GOOSE frame to its neighbors IEDs to inform them about the malfunctioning of CB-11. Fig. 8 shows the visualization of the generated trace file where the StNum increases from 1 to 2 and then to 3. On

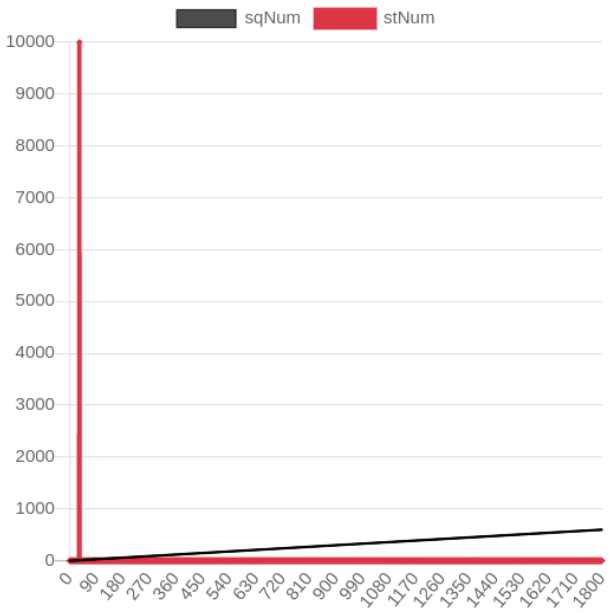

Fig. 9. Screenshot of attack induced trace with injection of high StNum of GOOSE frame.

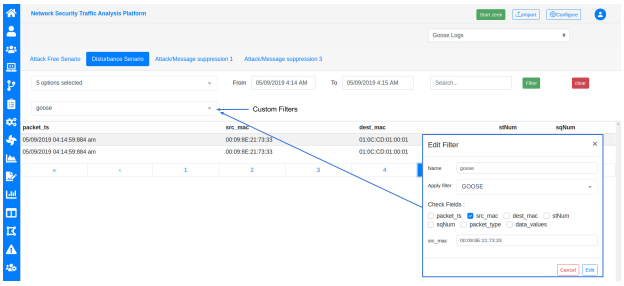

Fig. 10. NSTAP screenshot of Goose custom filter result.

the other hand, the SqNum resets to 0 when the disturbance happens. Finally, for the attack scenario, the attackers hijack the communication channel by injecting a GOOSE frame with a high StNum. Fig. 9 shows the screenshot of the attack trace with the high status number of 9999.

On the traffic logs side, as shown in Fig. 10, we can create custom filters for each log which will allow us to better analyze the traffic.

V. Future Work

Our future direction is to explore the collected data to work on intrusion detection using machine learning to introduce a framework dedicated to network security assessment and real time monitoring. The design and architecture of NSTAP allows us to collect enormous amounts of data which is an essential ingredient in machine learning. We intend to simulate different kinds of attacks in our laboratory's network. We will start with existing available datasets such as CIC-IDS 2017 [16] and IEC61850 SecurityDataset to build our ML model. Then we will test it on the traffic logs collected from the NIST SmartGrid testbed laboratory. The ability to import external datasets is important for machine learning training tools.

VI. Conclusion

In this paper, we presented the Network Security Traffic Analysis Platform with the objective to highlight its performance and scalability necessary to work with massive datasets.

The platform provides managed sophisticated probes and our motivation comes from the increase of attack attempts and the need to provide real time traffic analysis for our network. NSTAP provides great visualization tools and real-time filters and stream processing on the network traffic logs. We designed and implemented the NSTAP to be easy to use and deployed by dockerizing its services.

We demonstrated its value in an industrial network environment and were able to generate alerts and represent the traffic logs with meaningful graphs for an easy analysis.

References

[1] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, pp. 222–232, 1987.

[2] C. Liu and Y. Zhang, "An intrusion detection model combining signature-based recognition and two-round immune-based recognition." Institute of Electrical and Electronics Engineers Inc., 2021, pp. 497–501.

[3] M. Almseidin, M. Alzubi, S. Kovacs, and M. Alkasassbeh, "Evaluation of machine learning algorithms for intrusion detection system," in *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, 2017, pp. 000 277–000 282.

[4] T. Mehmood and H. B. Rais, "Machine learning algorithms in context of intrusion detection." Institute of Electrical and Electronics Engineers Inc., 12 2016, pp. 369–373.

[5] V. Bhatia, S. Choudhary, and K. Ramkumar, "A comparative study on various intrusion detection techniques using machine learning and neural network," in *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2020, pp. 232–236.

[6] A. Borkar, A. Donode, and A. Kumari, "A survey on intrusion detection system (ids) and internal intrusion detection and protection system (iidps)," in *2017 International Conference on Inventive Computing and Informatics (ICICI)*, 2017, pp. 949–953.

[7] C. Gkountis, M. Taha, J. Lloret, and G. Kambourakis, "Lightweight algorithm for protecting sdn controller against ddos attacks," in *2017 10th IFIP Wireless and Mobile Networking Conference (WMNC)*, 2017, pp. 1–6.

[8] A. Cirneci, S. Boboc, C. Leordeanu, V. Cristea, and C. Estan, "Netpy: Advanced network traffic monitoring," 2009, pp. 253–254.

[9] H. S. Qiu, W. Willinger, and J. Rexford, "Streaming data visualization for network security," 2017.

[10] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection: Methods, evaluation, and applications," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, 2003, pp. 234–247.

[11] D. Peng, Y. Qiao, and J. Yang, "Analyzing traffic characteristics between backbone networks based on hadoop." Institute of Electrical and Electronics Engineers Inc., 2014, pp. 149–153.

[12] Vern Paxson, "The zeek network security monitor," https://docs.zeek.org/en/master/about.html, 19985, accessed: 2022-08-06.

[13] Facebook, "osquery," https://osquery.readthedocs.io/en/latest/, 2014, accessed: 2022-08-10.

[14] C. Salter, "Future trends to smart grid automation architecture by iec 61850."

[15] P. P. Biswas, H. C. Tan, Q. Zhu, Y. Li, D. Mashima, and B. Chen, "A synthesized dataset for cybersecurity study of iec 61850 based substation." Institute of Electrical and Electronics Engineers Inc., 10 2019.

[16] Canadian Institute for Cybersecurity, "Intrusion detection evaluation dataset (cic-ids2017)," https://www.unb.ca/cic/datasets/ids-2017.html, 2017, accessed: 2022-08-05.