The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 2024

Volume 29, Pages 182-191

ICRETS 2024: International Conference on Research in Engineering, Technology and Science

A Graph Database Intrusion Detection and Prevention System

Simona Lyubenova
Sofia University "St. Kliment Ohridski"

Milen Petrov
Sofia University "St. Kliment Ohridski"

Adelina Aleksieva-Petrova
Technical University of Sofia

Abstract: Network threats are perceived as a serious and current problem due to the presence of different types of attacks, the purpose of which is to penetrate the security of a certain system using vulnerabilities and fraud techniques. They can appear anywhere, making them more difficult to detect and prevent. The victims of such type of attacks are constantly increasing, resulting in great losses not only in financial terms, but also in breaches of data privacy and business processes. As a result, protecting confidential information from unpredictable attacks has become a pressing issue and a difficult task that would be impossible without the help of intrusion detection systems (IDS) and intrusion prevention systems (IPS). The goal of the paper is to propose and design general architecture and implement a prototype for protection of an existing network of devices by detecting and preventing threats through the extraction and analysis of information from the devices located in the network, with the necessary data being stored in a graph database offering the possibility of visualization. To implement device network protection, it is necessary to enable software tools that, based on certain rules, impose restrictions on devices on the network and prevent future malicious actions.

Keywords: Graph database, Intrusion detection systems, Intrusion prevention systems, Network security

Introduction

In light of the increasing number and complexity of cyber threats in recent years, timely and efficient detection and prevention of malicious activity is crucial to safeguarding valuable systems and data (Rizvi, 2016). This underscores the need for early detection of cyberattacks. The prevalence of network threats is a major concern in current times, as various attacks aim to exploit vulnerabilities and use deceitful tactics to breach the security of systems. These threats can manifest in various forms and are challenging to identify and thwart.

Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) are widespread and commonly used (Thapa, 2020). In essence, Intrusion Detection and Prevention Systems (IDPS) are a security measure designed to protect networks from both external and internal threats. By monitoring network activity for suspicious patterns, IDPS can effectively detect and prevent cyberattacks (Birkinshaw, 2019).

The objective of this paper is to propose an architecture and design a graph-based system that can implement protection of an existing network of devices by detecting and preventing threats. The proposed NTDP (Network Threat Detection and Prevention) system implements a real-time collection and monitoring of network traffic, threats, and anomalies, and structure this information in a centralized, easily accessible location. The main contributions of the proposed architecture are (1) the capability of analyzing collected data to detect and prevent network attacks by activating detectors and implementing protective measures automatically and (2) to provide

a dynamic visualization of the network topology, using interactive graphical objects to display device attributes, statuses, and connections, with real-time updates reflecting any changes in the network structure or device conditions.

In order to apply protection to a network of devices, it is necessary to enable software tools that, based on certain rules, enforce restrictions on devices in the network and prevent future malicious actions. For simplicity, we can call these software tools by the short name "detectors". Detectors apply a set of rules on the previously available devices and on those that subsequently join the network, and along with this, some of the information stored in the graph database is modified in order to properly update the visualization.

The paper structure represents the following parts. The first part is introduction, defining the nature of the problem involved and formulates the set goals and tasks for the implementation of the research. The second part introduces related works, including approaches and methods for solving the problem. The third part, the focus falls on the general architecture and the design of its modules. The fourth part describes the experiments set up. The fifth part presents some results and discussion. The sixth part is final and includes a conclusion.

Related Work

To prevent cyber attacks, the analyze packet payloads against malware and intrusions is important part and IDS can accurately detect them. IDS is a process for monitoring and analyzing events that occur on a network and that pose a potential threat to it. IDS is responsible for monitoring data traffic on the network as well as any suspicious actions against its security. While IPS is a network monitoring process that is performed to prevent incoming threats and block ongoing attacks. IPS can be viewed as a combination of IDS and tools that respond to attacks that have occurred by implementing preventive measures (Thapa, 2020).

To address this, researchers are developed a taxonomy covers IDS architectures, detection methods, analysis techniques, responses, data sources, detection ranges, validation strategies, and performance metrics (Quincozes, 2021). Furthermore, authors have provided a comprehensive review of the detection rules used by the latest IDSs and evaluate their resilience against five types of attacks.

The algorithm by Turner (2016) provides an effective solution for determining the enabled and disabled states of rules in a signature-based IDS. By searching through rule sets and creating files to track the status of each rule, the algorithm allows for better management and monitoring of the rules within the IDS. This ultimately contributes to the overall effectiveness and efficiency of the IDS in detecting and responding to security threats.

Several previous studies have supported different approaches for IDPS. For examples, Software-Defined Networking (SDN) are used in IDPS design and implemention (Birkinshaw, 2019). This defense system constantly monitors network traffic for any abnormal or malicious behavior, and actively mitigates potential cyber threats. Also, some researches are developed an IDPS using SDN technology, designed to protect against ARP spoofing and Blacklisted MAC Addresses by adapting SDN's settings in real-time to identify and stop malicious network activity (Girdler, 2021).

To enhance security measures, a hybrid VM-based Honeypot system was implemented in conjunction with a hybrid IDPS (Rizvi, 2016). This setup compensates for any potential decrease in efficiency by focusing on signature-based methods for Network Intrusion Detection and Prevention Systems (NIDPS) and anomaly-based methods for Host Intrusion Detection and Prevention Systems (HIDPS), while still prioritizing the goal of minimizing resource consumption.

After analyzing the utilization of honeypots in corporate networks, virtualization technologies are implemented to decrease the costs associated with configuration, maintenance, and management (Baykara, 2018). The resulting system is an IDPS based on honeypots that visually displays network traffic on servers in real-time animations, allowing for easily accessible system information and the system is capable of detecting zero-day attacks through intrusion detection configuration.

The anomaly-based detection analyzes normal system behavior such as network packet information, operating system data, and system events (Sandhu, 2011). If behavior that differs from normal is observed or a potential threat is detected, the system generates an alert. Unusual activity that is not related to an attack is flagged as intrusive, which can result in a false alert being generated.

System Architecture

The NTDP intrusion detection and prevention system should have the capability to provide the following functionalities:

- Real-time collection of information on network traffic, identified threats and anomalies
- Structure the collected information and store it in a central location providing a means for easy access and management
- Providing secure access to the database through a user authentication and authorization system by managing permissions and roles to limit access to sensitive information
- Analyzing collected information to identify unwanted threats and anomalies with the ability to prevent network attacks by automatically activating detectors and implementing protective measures
- Developing or selecting algorithms to identify threats, detect anomalies and apply appropriate protective measures
- Visualizing the network topology through interactive graphical objects that reflect device attributes, their status, and the connections between them, and the ability to dynamically update as network structure or device status changes

Figure 1 presents an architectural view of the proposed NTDP system.

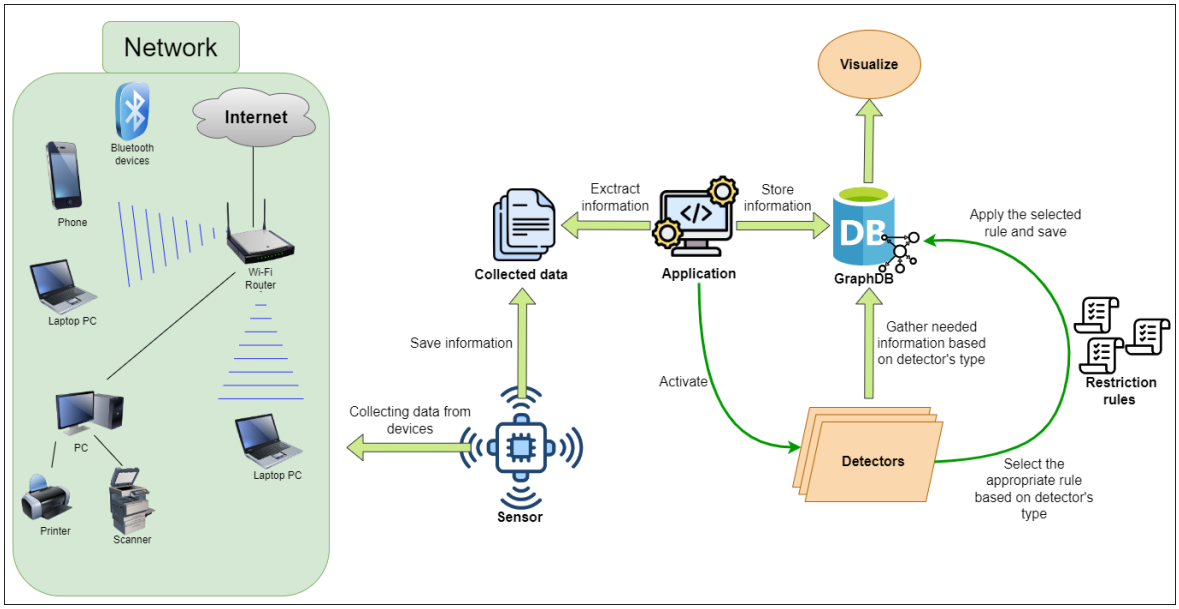


Figure 1. Developed architectural view of NTDP

An architecture is a collection of interacting components that are designed to perform certain tasks or functions (Figure 2).

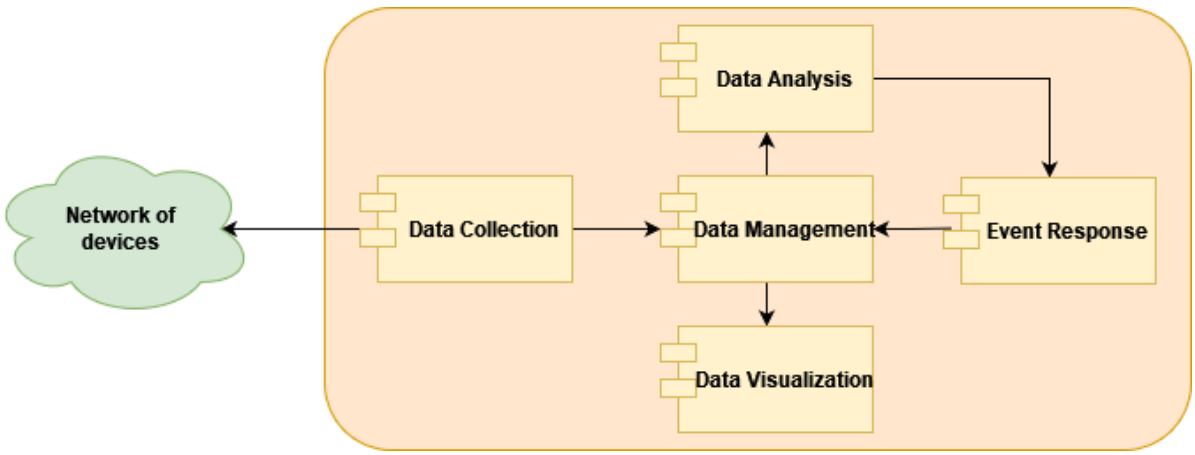


Figure 2. NTDP components architecture

Data Collection Component

The data collection component plays a key role in building an intrusion detection and prevention system. It is responsible for the active extraction, collection, aggregation and conversion into a structured format of information from various sources in the network such as hosts, switches, routers and other network devices. The collected information may include network traffic data, system events, active users, network device statistics, and other important security data.

The tasks of the data collection component include managing connections between network devices, extracting data from them, filtering and processing data, and transferring the collected information to a central storage location. The purpose of the data collection component is to provide a comprehensive overview of the network status and to provide the information necessary for security analysis and incident detection. For this reason, it is an essential element for the effective functioning of a system that includes monitoring and protection because it provides the basis for security-related decision making.

For the purpose of the study, we can assume that a network TAP (Test Access Point) sensor is used to collect information about devices in a network. It plays a key role in the development of the system by actively retrieving data about the network traffic and devices in the network. Similar to a TAP, the sensor allows the monitoring of network traffic without disconnecting devices. This ensures uninterrupted and unchanged access to the data, which is essential for quality system operation.

Data Management Component

The data management component provides the storage, organization and management of data collected from sensors or other sources. It is responsible for the central storage of the collected information and provides a convenient and efficient access to it. The data management component includes databases or other data storage mechanisms that allow for fast searching, filtering and processing of information. Data management is provided with functionalities such as adding, editing and deleting data, as well as their automated updating when necessary. The data management component may also include data protection mechanisms such as encryption and access control to ensure the security and confidentiality of stored information. The data management component in the development of the paper includes the selected graph database which provides various functionalities that satisfy the necessary requirements.

Data Analysis Component

The data analysis component provides functionalities to process, clarify and extract useful information from the collected data. It is designed to analyze the collected information to detect potential threats and anomalies in network behavior. The component applies various algorithms to detect anomalies, identify threats and extract important patterns from the data.

The data analysis component includes various analytical tools such as machine learning, statistical methods, graph algorithms that are used to process the information. It includes integration with other systems and modules that provide the necessary information to apply analysis. The data analytics component plays an important role in real-time event monitoring as well as incident response by providing important data for analysis to administrators to understand the current state of the network and take necessary measures to protect it.

The data analysis component in the development of the paper involves the activation of detectors that periodically monitor devices on the network at a specified interval and inspect them according to set criteria described in clearly defined detector rules. If the devices in the network match the set criteria, then the described response actions are applied by the activated detectors.

Data Visualization Component

The data visualization component aims to provide an intuitive and easy-to-understand way to visualize the network topology and device interactions on the network. It provides a graphical representation of the sensor data collected and processed by the other modules of the system. The data visualization component is connected

to the selected graph database, which contains as functionality the ability to visualize the stored data. It uses the device data in the network that is stored in the graph database to create a visual view of the connections and interactions between them. This can include connections between hosts, switches, routers, and other network components. Users are able to interact with the visualization to view details about individual devices, explore their connections, and analyze network data. This can include the ability to zoom in and out on portions of the graph, filter data to more easily discover key aspects of the network, select colors of different device types on the network, and options to add additional data and attributes to the visualization. The data visualization component provides a means to intuitively understand the network infrastructure and the interactions within it, which proves to be essential in order to quickly and effectively respond to potential threats and issues.

Event Response Component

The Event Response component is responsible for automated or manual action as a result of detected events or threats on the network. It has the ability to be configured to automatically respond to certain types of threats or events, which includes automatically blocking certain network connections, terminating services, or performing other actions to reduce the risk of attacks. Users have the ability to manually manage detected threats through the provided system visualization. Actions can include manually blocking selected network devices or connections, viewing detailed information about detected threats, and taking additional measures to prevent attacks. The event response component in the development of the study involves the activation of detectors that use clearly defined rules to respond if devices are detected that meet the set conditions described in the rules.

Setup of Experiments

For the purpose of the study, we will use off-the-shelf data collected from a network TAP sensor provided by a cybersecurity product development company, which will be used in the construction of the network topology. The devices present in the network are:

- a collection of computers, laptops, phones, and other devices that are under the common name Host;
- a set of network switches (switches), which are under the common name Switch;
- a collection of routers, which are collectively referred to as a Router.

In order to visualize a network topology with the collected data of devices of the listed types, it is necessary to load into the graph database CSV files that contain information about the fields of each object and information about the connections between the objects. The choice to represent objects, their attributes and relationships in the form of relational tables is to provide a clearer view of the properties of the objects and the relationships between them. Each table in Figure 3 represents a separate CSV file that contains the described fields. The graph database needs to be loaded with all CSV files to visualize the network topology. The Hosts, Switches, and Routers tables represent the nodes in the graph database, and the HostsToSwitches, SwitchesToRouters, and RoutersToRouters tables represent the connections between nodes in the graph database.

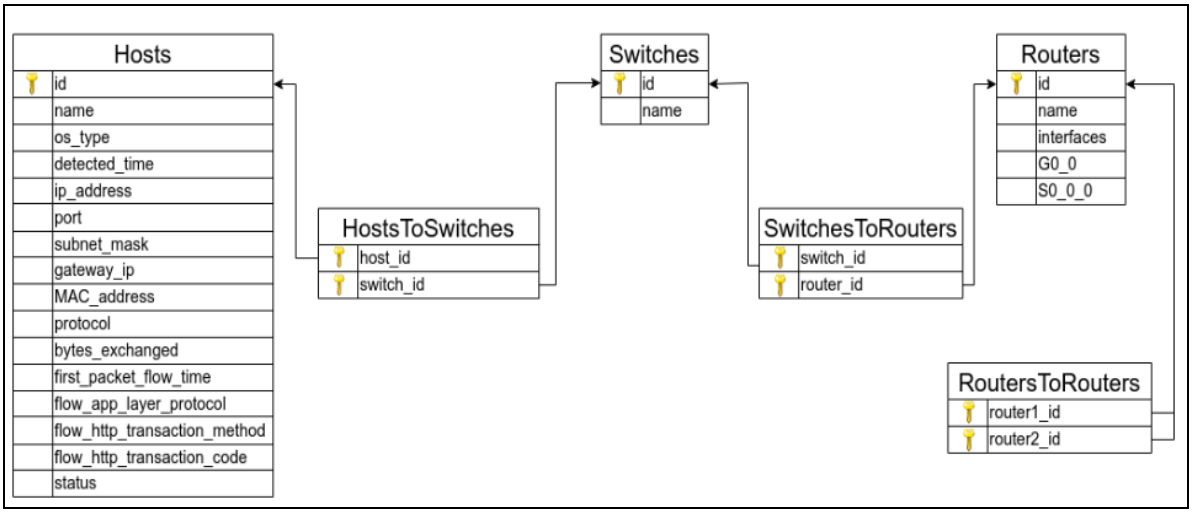


Figure 3. Representation of object types, their attributes and relationships as relational tables

Figure 4 presents some sample data for Host type devices to load into the graph base.

	A	B	C	D	E	F	G	H	I	J	K	L
1	id	name	os_type	detected_time	ip_address	port	subnet_mask	gateway_ip	MAC_address	protocol	bytes_exchanged	first_packet_flow_time
2	15969631	PC-12	WINDOWS	2024-04-01 05:53:24.000	192.168.1.2	59918	255.255.255.0	192.168.1.1	08:bb:64:3b:bc:1d	PROTO_TCP	0	2024-04-01 05:55:04.251
3	10185084	PC-13	MACOS	2024-04-01 07:01:13.000	192.168.1.3	135	255.255.255.0	192.168.1.1	32:ae:1e:2d:c7:7a	PROTO_TCP	2091	2024-04-01 07:01:22.593
4	20847490	PC-14	WINDOWS	2024-04-01 05:49:56.000	192.168.1.4	64427	255.255.255.0	192.168.1.1	ec:bf:6b:12:38:56	PROTO_TCP	15982	2024-04-01 05:51:44.840
5	20847491	PC-15	WINDOWS	2024-04-01 05:49:56.000	192.168.1.5	64427	255.255.255.0	192.168.1.1	60:49:10:b0:1b:da	PROTO_TCP	2781	2024-04-01 05:51:44.840
6	20848490	PC-16	LINUX	2024-04-01 05:45:54.000	192.168.1.6	63949	255.255.255.0	192.168.1.1	70:6a:ef:eb:74:b0	PROTO_TCP	0	2024-04-01 05:46:56.993
7	20847450	PC-17	WINDOWS	2024-04-01 05:45:54.000	192.168.1.7	63949	255.255.255.0	192.168.1.1	bf:97:da:d8:c7:1e	PROTO_TCP	20871	2024-04-01 05:46:56.993
8	20846490	PC-18	WINDOWS	2024-04-01 05:45:58.000	192.168.1.8	63961	255.255.255.0	192.168.1.1	5c:43:d0:45:2a:67	PROTO_TCP	0	2024-04-01 05:46:56.993
9	21847490	PC-19	MACOS	2024-04-01 05:45:58.000	192.168.1.9	63962	255.255.255.0	192.168.1.1	54:57:d3:cd:b3:42	PROTO_TCP	220	2024-04-01 05:46:56.993
10	20947490	PC-110	WINDOWS	2024-04-01 05:45:58.000	192.168.1.10	63963	255.255.255.0	192.168.1.1	71:75:77:42:20:c8	PROTO_TCP	0	2024-04-01 05:46:56.993
11	10183990	PC-111	LINUX	2024-04-01 05:48:05.000	192.168.1.11	54742	255.255.255.0	192.168.1.1	60:89:df:bf:12:48	PROTO_TCP	0	2024-04-01 05:48:25.073
12	10183999	PC-112	MACOS	2024-04-01 05:48:05.000	192.168.1.12	55742	255.255.255.0	192.168.1.1	45:ea:ab:96:78:6c	PROTO_TCP	8790	2024-04-01 05:48:25.073
13	11183990	PC-113	WINDOWS	2024-04-01 05:48:06.000	192.168.1.13	55748	255.255.255.0	192.168.1.1	71:58:7a:35:fe:d9	PROTO_TCP	0	2024-04-01 05:48:25.073
14	15183990	PC-114	WINDOWS	2024-04-01 05:48:06.000	192.168.1.14	55749	255.255.255.0	192.168.1.1	40:dd:ad:b1:1d:dc	PROTO_TCP	8920	2024-04-01 05:48:25.073
15	16183990	PC-22	LINUX	2024-04-01 05:48:06.000	192.168.2.2	55750	255.255.255.0	192.168.2.1	5c:71:20:d9:6f:16	PROTO_TCP	0	2024-04-01 05:48:25.073
16	17183990	PC-23	WINDOWS	2024-04-01 05:48:06.000	192.168.2.3	55751	255.255.255.0	192.168.2.1	c8:82:86:27:c5:03	PROTO_TCP	3245	2024-04-01 05:48:25.073
17	18183990	PC-24	WINDOWS	2024-04-01 05:48:06.000	192.168.2.4	55752	255.255.255.0	192.168.2.1	a2:8e:9e:f3:68:1b	PROTO_TCP	0	2024-04-01 05:48:25.073
18	19183990	PC-25	MACOS	2024-04-01 05:48:06.000	192.168.2.5	52753	255.255.255.0	192.168.2.1	d3:9e:e2:3e:46:6f	PROTO_TCP	1144	2024-04-01 05:48:25.073
19	10283990	PC-26	WINDOWS	2024-04-01 05:48:06.000	192.168.2.6	55756	255.255.255.0	192.168.2.1	5a:e1:55:41:0e:97	PROTO_TCP	3209	2024-04-01 05:48:25.073
20	10383990	PC-27	WINDOWS	2024-04-01 05:48:06.000	192.168.2.7	55757	255.255.255.0	192.168.2.1	08:52:79:12:bb:33	PROTO_TCP	0	2024-04-01 05:48:25.073
21	18602532	PC-28	LINUX	2024-03-30 05:38:09.000	192.168.2.8	59571	255.255.255.0	192.168.2.1	eb:d2:6d:7b:8f:ba	PROTO_TCP	3211	2024-03-30 05:38:09.000

Figure 4. Some sample data for Host devices

Sample data for Switch-type devices are presented in Figure 5 and Figure 6 presents sample data for Router-type devices.

	A	B
1	id	name
2	1	Switch-1
3	2	Switch-2

Figure 5. Representation of object types, their attributes and relationships as relational tables

	A	B	C	D	E
1	id	name	interfaces	G0_0	S0_0_0
2	1	Router-1	G0_0,S0_0_0	192.168.1.1/24	10.1.1.1/30
3	2	Router-2	G0_0,S0_0_0	192.168.2.1/24	10.1.1.2/30

Figure 6. Representation of object types, their attributes and relationships as relational tables

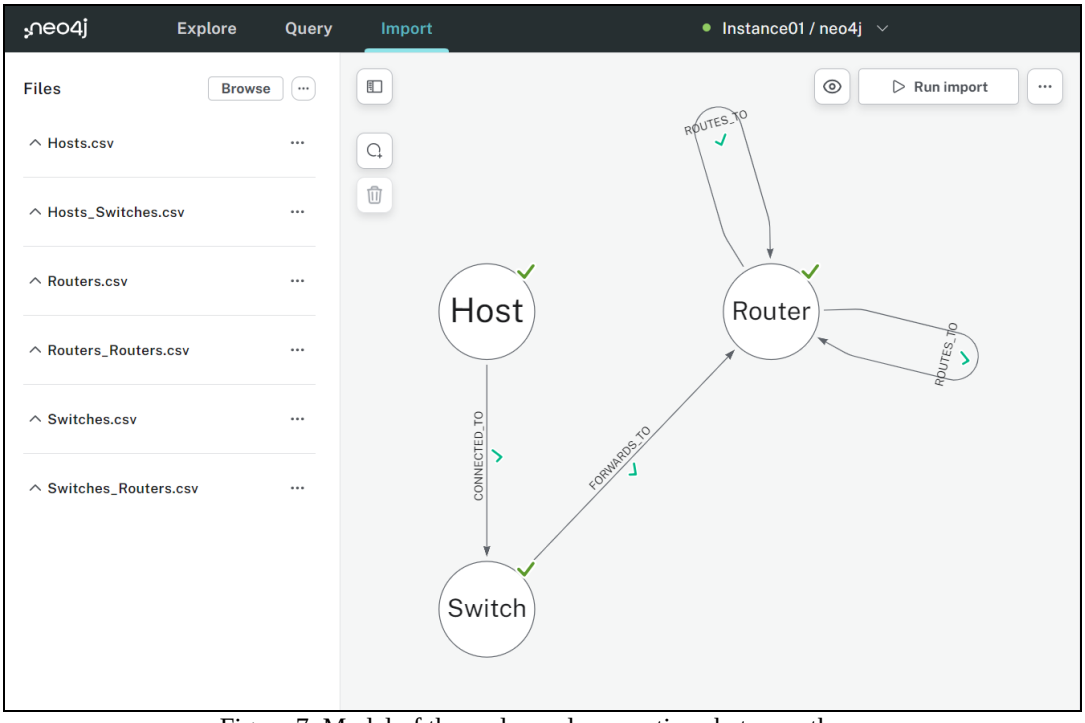


Figure 7. Model of the nodes and connections between them

The system is based on objects in Neo4j that need to be modelled against sample data. In the graph base, the model of the devices in the network infrastructure and their way of communication look like the one presented in Figure 7, which is a final view of the created nodes and their interconnections with collected data from loaded CSV files. A node labeled Router has two connections labeled ROUTES_TO to itself because in the created network, routers exchange information with each other and its transmission is bidirectional.

Rows to nodes and tables to label names:

- Each row from the Hosts relational table and the Hosts.csv file becomes a node in our graph labeled Host.
- Each row from the Switches table and from the Switches.csv file becomes a node in our graph labeled Switch.
- Each row from the Routers table and from the Routers.csv file becomes a node in our graph labeled Router.

Connections between nodes:

- The connection between nodes labeled Host and Switch is done with the relational table HostsToSwitches, represented as the file Hosts_Switches.csv, which is a connection labeled CONNECTED_TO.
- The connection between nodes labeled Switch and Router is accomplished with the SwitchesToRouters relational table, represented as the file Switches_Routers.csv, which is a relationship labeled FORWARDS_TO.
- The connection between nodes with Router and Router labels is implemented with the RoutersToRouters relational table, represented as the Routers_Routers.csv file, which is a relation with the ROUTES_TO label.

The data analysis component and the event response component are key in the development of an intrusion detection and prevention system. In a network, it is necessary to analyze information about the devices in the network and detect potential threats and anomalies in the network behavior that are present, against which protective measures can be taken to respond to the emerging threats. In this research, the two components are unified by the use of detectors that load specific rules from yaml files to filter the data of devices in the network according to set criteria and take actions towards them. The final step aims to simulate a working intrusion detection and prevention system. In the beginning, a network topology is created from devices with ready data loaded. We will use the network topology from Figure 8.

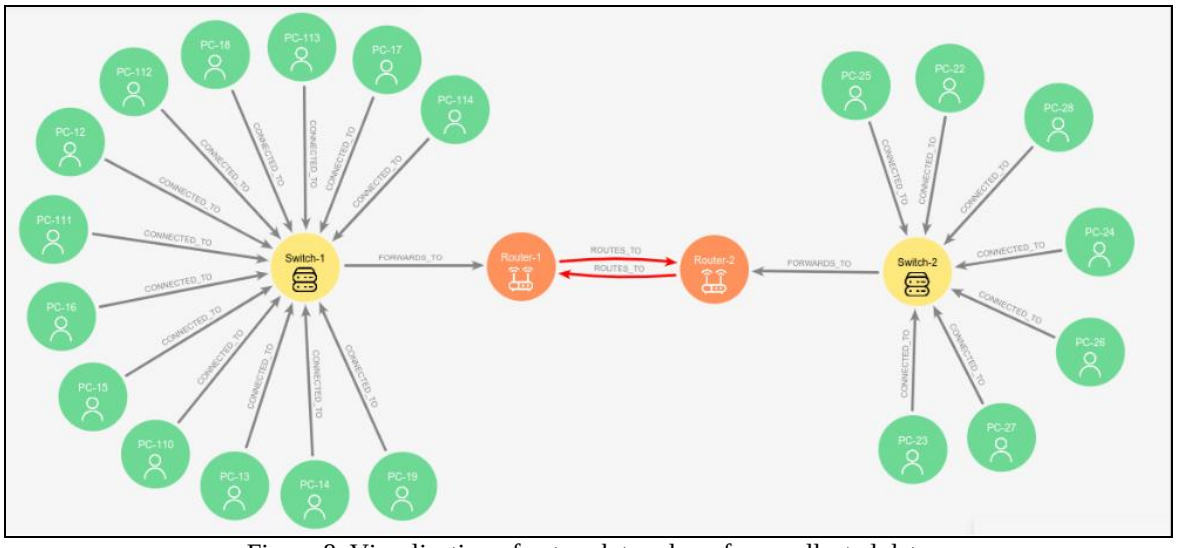


Figure 8. Visualization of network topology from collected data

Results and Discussion

When the simulation starts, devices from an additional CSV file that contains data for devices outside the network begin connecting to the network periodically. In parallel, both types of detectors are activated, which load their intended rules and apply restrictions on devices that are on the network as well as those that are

joining it. To accomplish this, two threads are created, one to add devices to the network and one to periodically activate the detectors. Results of the simulation execution can be observed through the outputted logs as well as the visualization (Figure 9). The displayed logs help to track the actions on the network if there is a problem with the preview or if a moment is missed.

```
Activated detector AppLayerProtocolDetector: Host-10185261,PC-211,FREE->SUSPEND.
Connected Host-23879925,PC-116.
Packet check before sending: Host-20947490,PC-110,SUSPEND :---> Host-23879925,PC-116,FREE.
Packet check before sending: Host-20947490,PC-110,SUSPEND <---: Host-23879925,PC-116,FREE.
Packet not sent: Host-10185084,PC-13,BLOCK x---> Host-23879925,PC-116,FREE.
Packet not sent: Host-10185084,PC-13,BLOCK <---x Host-23879925,PC-116,FREE.
Activated detector PortDetector: Host-23879925,PC-116,FREE->BLOCK.
Deactivated detector AppLayerProtocolDetector: Host-15969631,PC-12,SUSPEND->FREE.
Deactivated detector AppLayerProtocolDetector: Host-10185084,PC-13,BLOCK->FREE
```

Figure 9. A sample of the logs from the execution of the simulation

A visualization of the network topology during the simulation is presented in Figure 10. The color of devices labeled Host is determined by their status - green devices are in FREE status, red devices are in BLOCK status, and the rest are in SUSPEND status. The status change is carried out by the activated detectors, which apply actions from the loaded rules according to the criteria described in them. The devices with the FREE status are those that do not threaten the network in relation to the activated detectors, and the rest are those that in some way pose a threat to it. Figure 11 (a, b) shows a data visualization for devices with a BLOCK state enforced by both types of detectors, and Figure 11 (c) shows a data visualization for a device with a SUSPEND state enforced by the AppLayerProtocolDetector.

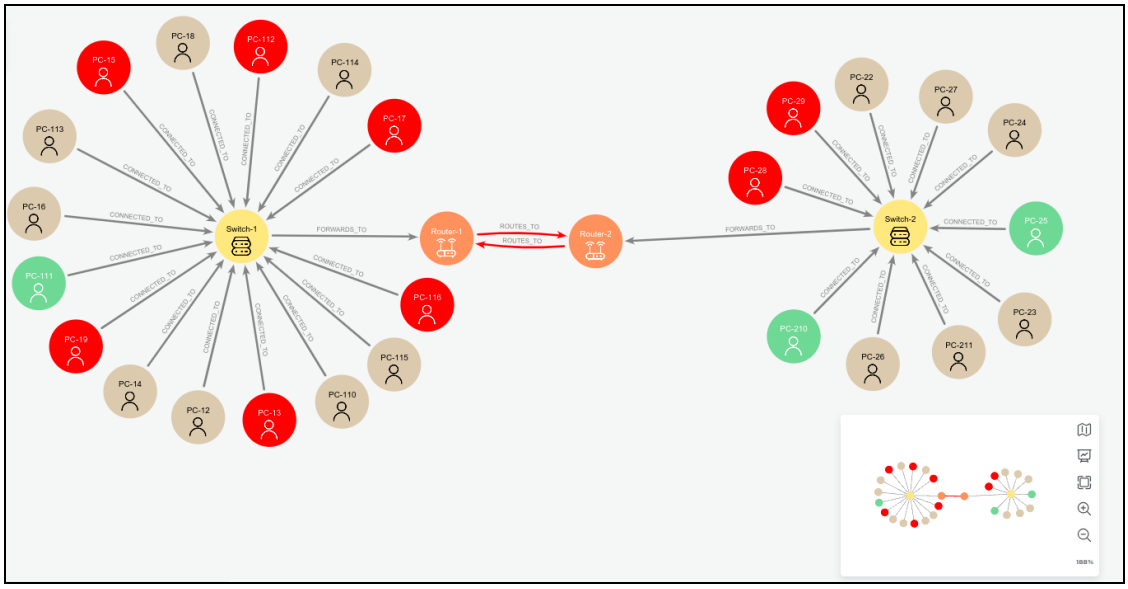
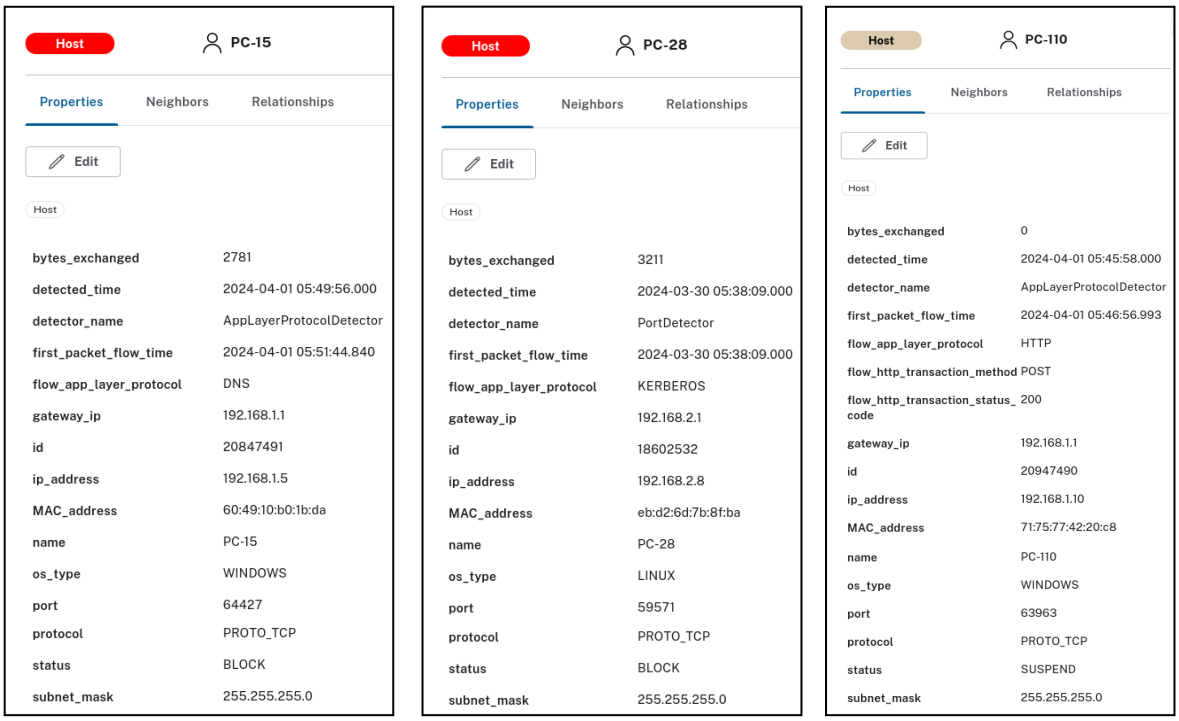


Figure 10. Visualization of the network topology during simulation execution for a working system

A new detector_name attribute has been added, indicating the name of the detector that applied a rule on the corresponding device. Both devices are of BLOCK status, the main difference between them being the values of the detector_name attribute - one device is limited by AppLayerProtocolDetector and the other is limited by PortDetector.

The status of the device in Figure 11 (c) is SUSPEND, the difference with the devices in Figure 10 being the changed color after applying rules by detectors. The detector that applied constraints to PC-110 in Figure 11 (c) is the AppLayerProtocolDetector that also applied constraints to PC-15 in Figure 11 (a). The detector applied different constraints to the two devices and the main difference between them is the status and changed color after the constraints are applied. The preview offers the possibility to observe the created network and to manually impose changes on it. The technologies used lay the foundations for future scalability and portability of the system. The developed modules are designed so that they can be easily extended by adding new functionalities to them. Visual and log-tracing testing has been performed to show that the system meets the requirements and is ready for use.



(a) (b) (c)

Figure 11. Data preview for devices labeled Host with (a, b) BLOCK status and (c) SUSPEND

The NTDP intrusion detection and prevention system is scalable, i.e. the system supports the addition and removal of devices as well as network expansion without loss of efficiency and performance. As a result, the system can process a minimum of 10,000 events per second when new devices are added or when the network expands, and keeps the event processing time below 1 millisecond regardless of the number of connected devices in the network. Performance metrics include monitoring resource usage such as CPU and memory under varying network load and expansion. In terms of efficiency, the system provides a quick response in data processing and threat response. Performance metrics include measuring system response time when operations such as adding new devices to the network or when a change occurs to the network. For example, when adding a new device to the network, a response time of less than 1 second was achieved, and the maximum allowable response time was 3 seconds.

A system using enabled detectors monitors the behavior of devices on a network and implements protective measures using restriction rules to prevent future malicious actions. This results in improved network security and provides continuous control and protection against potential threats from both current and those devices that will subsequently join the network. The construction of such a system represents an initial step, providing basic mechanisms for data processing and application of protective measures. This initiative serves as a foundation for further development and refinement of network security. The expected benefits of the implementation include better control and visibility over network activities, which will lead to greater stability and reliability of the built infrastructure through fewer work interruptions, faster recovery from incidents and better management of security risks to prevent unwanted events and losses from potential network threats.

Conclusion

This paper proposed a general architecture and presented the individual modules in the system (data collection module, data management module, data analysis module, data visualization module and event response module) and data model. As a result, objects with their attributes and relations were modeled and loaded with data in the graph base for building a network topology. The originally set objectives for the implementation of the present experiment have been fulfilled. Sample device data has been collected, using which a sample network topology has been built. The data is stored in a graph database, providing the possibility of visualization. Functionalities have been created to join a new device to the available network, to remove an existing device and its connections, and to change properties of an existing device. The system offers options for activating and deactivating detectors that monitor the behavior of devices in the network according to the loaded rules and impose restrictions on the threatening part of them.

The performed simulation shows that the developed system based on the proposed intrusion detection and prevention architecture works according to our expectations, being able to load a large volume of data in real time, which helps to effectively detect and prevent threats. Both visual results of it and results in the form of logs are provided. The system meets the requirements set for it and provides the necessary results.

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM journal belongs to the authors.

Acknowledgements

* This article was presented as a poster presentation at the International Conference on Research in Engineering, Technology and Science (www.icrets.net) held in Thaskent/Uzbekistan on August 22-25, 2024.

* The research reported here was funded by the project "Research and application of machine learning algorithms in the analysis and development of highly secure software", fund with contract KP-06-N57/4 from 16.11.2021 by Bulgarian National Science.

References

Baykara, M., & Das, R. (2018). A novel honeypot based security approach for real-time intrusion detection and prevention systems. *Journal of Information Security and Applications, 41, 103*-116.

Birkinshaw, C., Rouka, E., & Vassilakis, V. G. (2019). Implementing an intrusion detection and prevention system using software-defined networking: Defending against port-scanning and denial-of-service attacks. *Journal of Network and Computer Applications*, *136*, 71-85.

Girdler, T., & Vassilakis, V. G. (2021). Implementing an intrusion detection and prevention system using Software-Defined Networking: Defending against ARP spoofing attacks and Blacklisted MAC Addresses. *Computers & Electrical Engineering*, *90*, 106990.

Quincozes, S. E., Albuquerque, C., Passos, D., & Mossé, D. (2021). A survey on intrusion detection and prevention systems in digital substations. *Computer Networks*, *184*, 107679.

Rizvi, S., Labrador, G., Guyan, M., & Savan, J. (2016). Advocating for hybrid intrusion detection prevention system and framework improvement. *Procedia Computer Science*, *95*, 369-374.

Sandhu, U. A., Haider, S., Naseer, S., & Ateeb, O. U. (2011). A survey of intrusion detection & prevention techniques. In 2011 *International Conference on Information Communication and Management*, IPCSIT, *16*, 66-71.

Thapa, S., & Mailewa, A. (2020, April). The role of intrusion detection/prevention systems in modern computer networks: A review. In *Conference: Midwest Instruction and Computing Symposium (MICS)*, *53*, 1-14.

Turner, C., Jeremiah, R., Richards, D., & Joseph, A. (2016). A rule status monitoring algorithm for rule-based intrusion detection and prevention systems. *Procedia Computer Science*, *95*, 361-368.

Author Information

Simona Lyubenova
Sofia University "St. Kliment Ohridski"
5, James Bourchier Blvd., Sofia, Bulgaria

Milen Petrov
Sofia University "St. Kliment Ohridski"
5, James Bourchier Blvd., Sofia, Bulgaria
Contact e-mail: *milenp@fmi.uni-sofia.bg*

Adelina Aleksieva-Petrova
Technical University of Sofia
8, Kliment Ohridski St., Sofia, Bulgaria

To cite this article:

Lyubenova, S., Petrov, M. & Aleksieva-Petrova, A. (2024). A graph database intrusion detection and prevention system. *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 29*, 182-191.