Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Real-Time Anomaly Detection in Software-Defined Networks using an Integrated ELK Stack and Machine Learning Framework

Mustapha ZEROUAL

zeroual.mustapha@ucd.ac.ma

Chouaib Doukkali University

Youssef BADDI

Chouaib Doukkali University

Anass SEBBAR

International University of Rabat

Faycal BENSALAH

Chouaib Doukkali University

Research Article

Keywords: Software-Defined Networking, Anomaly Detec-tion, ELK Stack, Machine Learning, Network Security, Log Analysis

Posted Date: November 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10634218/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Real-Time Anomaly Detection in Software-Defined Networks using an Integrated ELK Stack and Machine Learning Framework

Mustapha Zeroual*, Youssef Baddi* , Anass Sebbar†, ,Faycal BENSALAH *
*STIC Lab, FSJ, Chouaib Doukkali University, El Jadida, 24000, Morocco
†TICLAB,International University of Rabat, RABAT 10000
, Morocco
Corresponding author email:zeroual.mustapha@ucd.ac.ma

***Abstract*—Software-Defined Networking (SDN) enhances network programmability and management by separating the control and data planes. However, this centralized architecture introduces significant security vulnerabilities, making the controller a prime target and necessitating robust, real-time monitoring. Traditional tools often fall short in dynamic SDN environments. This paper presents an integrated framework for real-time anomaly detection in SDN. The system utilizes Mininet for SDN emulation with an OpenDaylight controller, and leverages the ELK stack [6] (Elasticsearch, Logstash, Kibana), augmented by Filebeat and RSyslog, for comprehensive log collection from both control and data planes. Logs are pro-cessed, parsed, and indexed for analysis. Machine learning capabilities within Kibana are employed for anomaly detection. Experimental results demonstrate successful centralized log management, real-time visualization of network activity, and the detection of 47 distinct anomaly events during simulated DDoS-like attacks, with these events further classified into 50 critical/major and 28 warning severity instances. The primary contribution is a practical, end-to-end system demonstrating effective real-time anomaly detection in SDN using a customized ELK and ML approach.**

***Index Terms*—Software-Defined Networking, Anomaly Detection, ELK Stack, Machine Learning, Network Security, Log Analysis.**

1. Introduction

Software-Defined Networking (SDN) has emerged as a transformative networking paradigm, fundamentally altering network management and operation by decoupling the control plane from the data plane. This separation enables centralized network intelligence and programmability, offering benefits such as enhanced flexibility, scalability, and agile service deployment [1,2] The SDN controller, acting as the network's "brain," provides a global view and facilitates dynamic configuration of network resources. However, the very centralization that empowers SDN also introduces significant security challenges. The controller becomes a critical single point of failure and a lucrative target for attackers; its compromise can lead to widespread network disruption. Traditional network monitoring tools, often designed for distributed architectures, struggle with the dynamic nature and unique log sources of SDN environments. Efficient real-time log analysis and proactive anomaly detection are therefore paramount for securing SDN infrastructures. The challenge lies in designing an architecture that can centralize and process logs from diverse SDN components controllers[3], switches, and hosts—and apply intelligent analytics to detect malicious activity in real time, without requiring deep in-house networking expertise for day-to-day operations. This centralized control in SDN, while beneficial, necessitates a correspondingly robust and intelligent centralized security oversight mechanism [4,5].

To address these challenges, this paper proposes an integrated system that leverages the ELK stack (Elasticsearch, Logstash, Kibana) along with Mininet for network emulation, OpenDaylight (ODL) as the SDN controller, Filebeat, and RSyslog. This framework provides a comprehensive solution for log collection from both control and data planes, real-time processing and analysis, interactive visualization, and machine learning (ML)-based anomaly detection tailored for SDN environments. While the ELK stack provides a robust foundation for log management, it requires substantial customization to function effectively as a Security Information and Event Management (SIEM) tool, particularly in specialized domains such as SDN. This work focuses on such customization, including specific log parsing strategies and the integration of ML within Kibana for detecting anomalous patterns indicative of attacks.

The key contributions of this work are:

- The design and implementation of an end-to-end framework for real-time aggregation, processing, and analysis of logs from both control (OpenDaylight) and data planes (Open vSwitch) in an SDN environment using a customized ELK stack.
- The demonstration of a machine learning mechanism integrated within Kibana for effective anomaly detection, specifically identifying patterns indicative of Distributed Denial of Service (DDoS) attacks, based on processed SDN logs.
- Empirical validation of the system's capabilities through simulation, showcasing comprehensive log visualization and quantitative anomaly detection metrics.

The remainder of this paper is structured as follows:

Section II discusses related work. Section III details the system design and methodology. Section IV presents the experimental setup and results. Section V offers a brief discussion of the findings and limitations. Finally, Section VI concludes the paper and suggests directions for future research.

2. baground and Related Work

This section reviews existing research in SDN security monitoring, the use of the ELK stack for network security, and machine learning approaches for anomaly detection in SDN.

2.1. Security Monitoring in SDN

The centralized architecture of SDN, while offering numerous advantages, presents unique security vulnerabilities. The SDN controller is a tempting attack surface [7], and its compromise can have catastrophic consequences for the entire network. Other challenges include vulnerabilities in the OpenFlow protocol, potential deficiencies in communication verification between planes, and network resource constraints [8,9]. Effective network measurement is crucial for understanding network behavior, facilitating monitoring, enabling anomaly detection, and establishing robust security mechanisms. Monitoring approaches in SDN often involve flow-based analysis, statistical data collection from network elements, and topology discovery. OpenDaylight, a popular open-source SDN controller, provides APIs and modules that can be leveraged for network monitoring and statistics collection, forming a basis for security applications [10].

2.2. ELK Stack for Network Security and SIEM

The ELK stack, comprising Elasticsearch, Logstash, and Kibana, along with Beats for data shipping, is a widely adopted open-source solution for log management. It excels at collecting logs from diverse sources, processing (parsing and transforming) them, storing and indexing them for efficient search in Elasticsearch, and visualizing them through Kibana dashboards. While powerful for log aggregation and analysis, the ELK stack in its native form is not a full-fledged SIEM. It often lacks advanced, out-of-the-box capabilities such as sophisticated event correlation, automated alerting, and comprehensive incident management functionalities found in commercial SIEMs. Customization, including the development of specific parsing rules (e.g., using Grok for Logstash) and the integration of additional tools or plugins for alerting and ML, is typically required to tailor ELK for specific security monitoring use cases. The project detailed in this paper undertakes such customization to build SIEM-like capabilities for SDN.

2.3. Machine Learning for Anomaly and DDoS Detection in SDN

Machine learning techniques are increasingly being applied to detect anomalies and cyberattacks, including DDoS attacks, in network traffic. In SDN environments, ML models such as Support Vector Machines (SVM), Naive Bayes, Multi-Layer Perceptrons (MLP), Neural Networks, and Random Forest have shown promise in distinguishing malicious traffic from benign traffic by analyzing flow statistics or packet features. For instance, MLPs have demonstrated high accuracy in classifying TCP-SYN flood traffic in simulated SDN environments [11, 12, 13]. However, the effectiveness of ML-based detection is often hampered by challenges such as the availability of high-quality, representative datasets, the need for real-time processing at large network scales, the computational complexity of some models, and overall resource efficiency. Simulation environments like Mininet are often used to generate controlled datasets for training and testing these ML models when real-world SDN attack data is scarce or sensitive [14,15]. This approach, while not perfectly replicating real-world conditions, allows for the systematic evaluation of detection mechanisms.

2.4. Positioning This Work

While individual components like SDN monitoring, ELK-based log analysis, and ML for anomaly detection are established research areas, this paper focuses on their practical, engineered synergy within a cohesive framework. The contribution lies in the detailed design and end-to-end implementation of a system that collects heterogeneous logs from both SDN control and data planes, employs tailored parsing for these specific log types, and integrates an ML-based anomaly detection mechanism directly within the Kibana visualization platform. This demonstrates how the ELK stack can serve as an enabling data pipeline for effective ML-based security monitoring in a simulated SDN environment, moving beyond isolated component testing to a holistic system validation.

3. System Design and Methodology

This section details the architecture of the proposed SDN anomaly detection system, the log collection and processing pipeline, and the machine learning-based detection mechanism.

3.1. System Architecture

The overall architecture, depicted in Fig. 1, integrates several key components to achieve real-time anomaly detection in an SDN environment.

- **SDN Environment**: Mininet is used to emulate a network topology consisting of OpenFlow-enabled switches (Open vSwitch - OVS) and hosts. An OpenDaylight (ODL) instance acts as the remote SDN controller, managing network flows and policies.
- **Log Collection**: Filebeat, a lightweight log shipper, is deployed to collect logs. It directly tails logs from the ODL controller. For data plane logs from OVS, RSyslog is used to centralize them initially, after

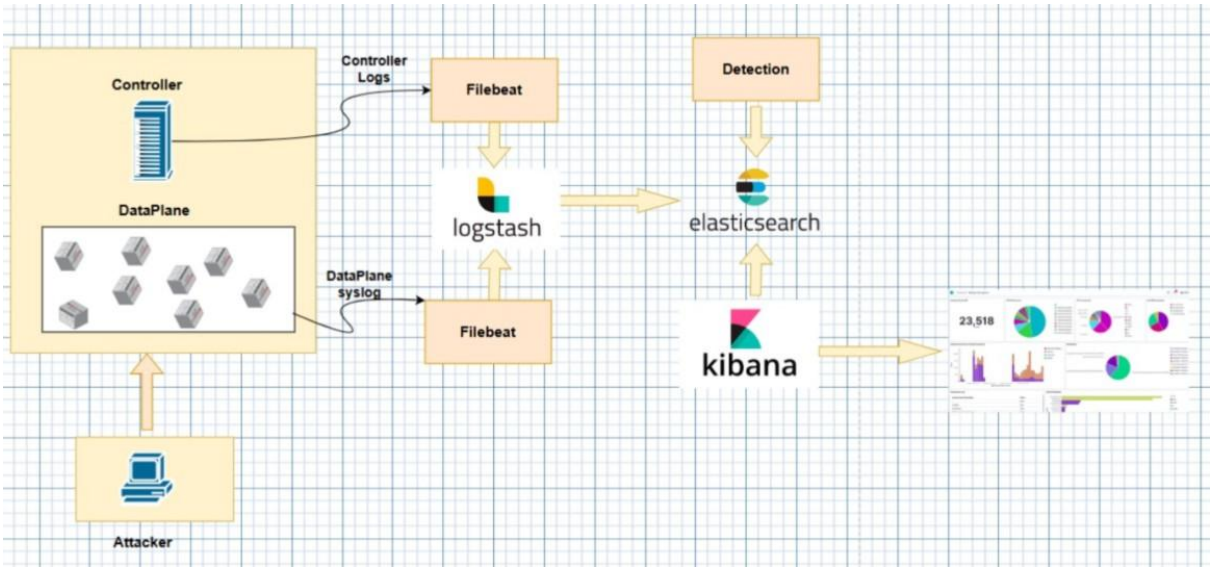


Figure 1. System Architecture for SDN Anomaly Detection. Logs from the SDN Controller (OpenDaylight) and Data Plane (Mininet/Open vSwitch) are collected by Filebeat (directly or via RSyslog), processed by Logstash, stored in Elasticsearch, and analyzed/visualized in Kibana, which also hosts the ML detection module.

which Filebeat collects them for forwarding. This differentiated approach caters to the distinct logging mechanisms of control and data plane components.

- **Log Processing & Storage**: Logstash ingests logs from Filebeat, parses them using Grok patterns to transform unstructured data into structured JSON, and enriches them with relevant metadata. Elasticsearch then stores and indexes these structured logs, enabling fast querying and analysis.
- **Analysis, Visualization & Detection**: Kibana provides a web interface for querying logs, creating dashboards for real-time visualization of network activity, and implementing machine learning jobs for anomaly detection.

This architecture facilitates a continuous flow of log data from generation points within the SDN to the analysis and detection engine.

3.2. Log Collection and Processing

Effective log collection and processing are critical for meaningful analysis and accurate anomaly detection. The system employs distinct strategies for control and data plane logs.

3.2.1. Control Plane Log Collection (OpenDaylight). Logs from the OpenDaylight controller, which reside in its Karaf directory (e.g., data/log/karaf.log), provide insights into controller operations, application behavior, and potential security events targeting the control plane. Filebeat is configured via its filebeat.yml file to monitor these log files. Key configurations include specifying the log paths and adding a custom field, such as log_type: controller, to distinguish these logs during processing. Filebeat then forwards these logs directly to Logstash.

3.2.2. Data Plane Log Collection (Open vSwitch). Data plane logs from Open vSwitch offer visibility into packet forwarding behavior, flow rule matching, and traffic statistics. Two OVS logging modules, dpif (for packet processing details) and openflow (for OpenFlow message exchanges), are activated. To capture detailed packet handling information, the dpif module is set to debug mode using the command sudo ovs-appctl vlog/set dpif:DBG. OVS logs are first centralized using RSyslog, which acts as a local aggregation point. Filebeat is then configured to collect these logs from RSyslog's designated output files, tagging them with a field like log_type: dataplane. This two-step process for OVS logs accommodates standard syslog outputs from network devices and allows for potential pre-filtering or buffering by RSyslog if needed in more complex setups.

3.2.3. Log Parsing and Indexing (Logstash & Elasticsearch). Raw logs from both control and data planes are often unstructured or semi-structured. Logstash, listening on port 5044 for incoming Beats data, plays a crucial role in transforming these logs into a structured format suitable for analysis. Grok patterns are defined within the Logstash configuration to parse specific log formats based on the log_type field assigned by Filebeat. These patterns extract relevant fields such as timestamps, source/destination IPs, port numbers, log levels, and messages into a JSON structure. The quality of this parsing stage is paramount, as inaccurate or incomplete parsing would lead to flawed data in Elasticsearch, undermining subsequent analysis and the reliability of the ML model. Once parsed and structured, Logstash forwards the data to Elasticsearch, which indexes

the logs on localhost:9200, making them searchable and available for Kibana.

3.3. Anomaly Detection Mechanism

The system leverages Kibana's built-in machine learning capabilities for anomaly detection. Specifically, a "unique count" or "high count" job type is configured to monitor metrics indicative of DDoS attacks, such as an abnormally high number of requests or connections observed in the processed logs within a given time window. This approach focuses on identifying deviations from established baselines of network behavior. The ML job is trained on the ingested log data, learning normal patterns to subsequently flag significant deviations as anomalies. Detection rules can be set up in Kibana to generate alerts when anomalies surpassing certain severity thresholds are identified by the ML model.

4. Experimental Evaluation

This section describes the experimental setup used to validate the proposed system and presents the results obtained from simulated attack scenarios.

4.1. Experimental Setup

4.1.1. Network Topology. The SDN environment was emulated using Mininet. The topology consisted of five OpenFlow-enabled switches (simulated by Open vSwitch) and four hosts connected to each switch. This network was managed by a remote OpenDaylight (Karaf) controller, accessible at IP address 192.168.114.137 on port 6653. Initial connectivity tests, using ping operations between all host pairs, confirmed that the network topology was correctly configured and all components were communicating successfully.

4.1.2. ELK Stack Configuration. The ELK stack components were configured as follows:

- **Filebeat**: Monitored ODL logs from /data/log/karaf.log and OVS logs centralized by RSyslog. Custom fields (log_type: controller or log_type: dataplane) were added to differentiate log sources.
- **Logstash**: Listened on port 5044 for data from Filebeat. Employed Grok patterns tailored to parse ODL and OVS log formats.
- **Elasticsearch**: Received structured logs from Logstash and indexed them on localhost:9200.
- **Kibana**: Accessed data from Elasticsearch via localhost:5601. Used for log visualization, dashboard creation, and configuring the machine learning job for anomaly detection.

4.2. Attack Scenario and Results

The system's efficacy was tested by simulating network attacks and observing the detection capabilities.

TABLE 1. ANOMALY DETECTION PERFORMANCE SUMMARY

Metric	Value
Total Distinct Anomaly Events Identified	47
Instances Classified as Critical/Major Severity	50
Instances Classified as Warning Severity	28

4.2.1. Simulated Attack. The primary focus was on detecting DDoS-like behavior. The machine learning job in Kibana was configured to monitor for an abnormal number of requests or network events, which is a common characteristic of DDoS attacks. An attempt to simulate a macof attack (MAC flooding) was made, but its high computational requirements on the test setup led to its early termination to prevent system overload. The results presented primarily pertain to the detection of volumetric anomalies based on request patterns.

4.2.2. Log Visualization and Analysis. Kibana dashboards provided real-time visualization of ingested logs, clearly differentiating between control plane (ODL) and data plane (OVS) logs through distinct representations (e.g., color-coded bars). This allowed for qualitative assessment of network activity. Analysis of individual log entries in Kibana showcased the structured data extracted by Logstash. For instance, control plane logs displayed fields such as log_source: controller, input.type: log, and detailed messages regarding controller events. Similarly, data plane logs revealed crucial traffic information, including source and target IP addresses, MAC addresses, and port numbers, enabling the tracing of network communications. This end-to-end visibility from log generation through parsing to visualization validates the integrity of the data pipeline feeding the ML detection mechanism.

4.2.3. Quantitative Anomaly Detection Results. The machine learning model integrated within Kibana successfully identified anomalous network behavior during the simulated attacks. As summarized in Table 1, the system flagged a total of 47 distinct anomaly events. Further classification by the ML model based on severity indicated that these events corresponded to 50 instances of critical/major severity and 28 instances of warning severity. This severity breakdown, determined by the ML model's assessment of log patterns associated with these events, provides actionable intelligence, allowing security analysts to prioritize the most significant threats. This capability moves beyond simple binary detection (anomaly/normal) to offer a more nuanced understanding of potential security incidents, which is crucial for effective response in a Security Operations Center (SOC) environment.

Figure 2 presents a dashboard that summarizes SDN traffic and security events during the attack simulation. The visible peaks indicate abnormal increases in activity relative to baseline behavior, enabling rapid detection of when the attack starts and how traffic intensity evolves.

Figure 3 illustrates this view, showing the raw SDN logs after collection, indexing, and parsing in ELK. It

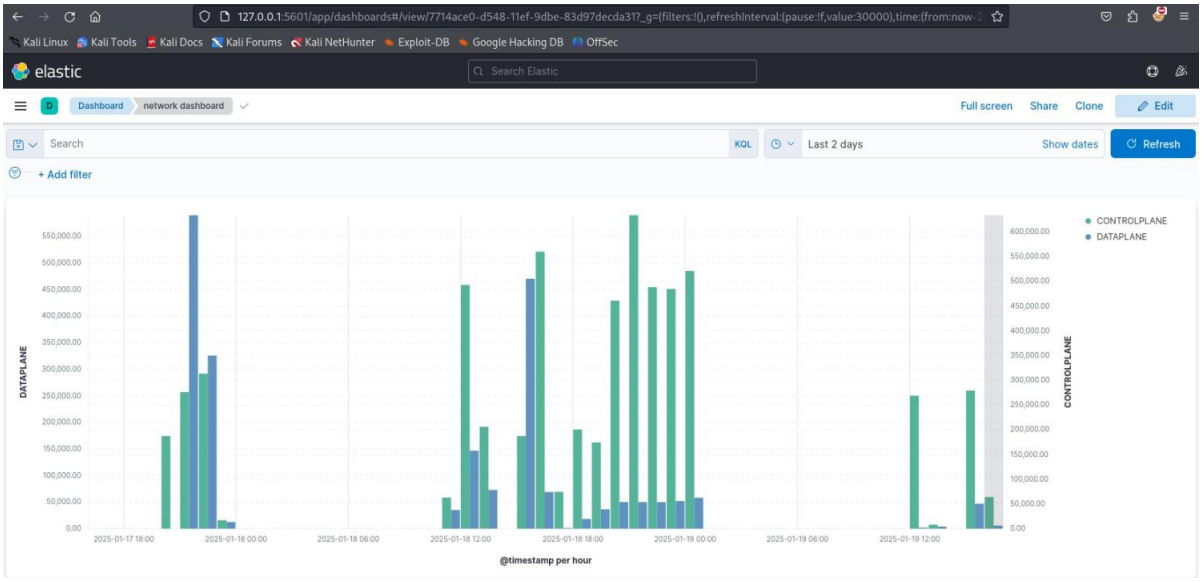


Figure 2. Kibana Dashboard: SDN Traffic Overview During Attack Simulation

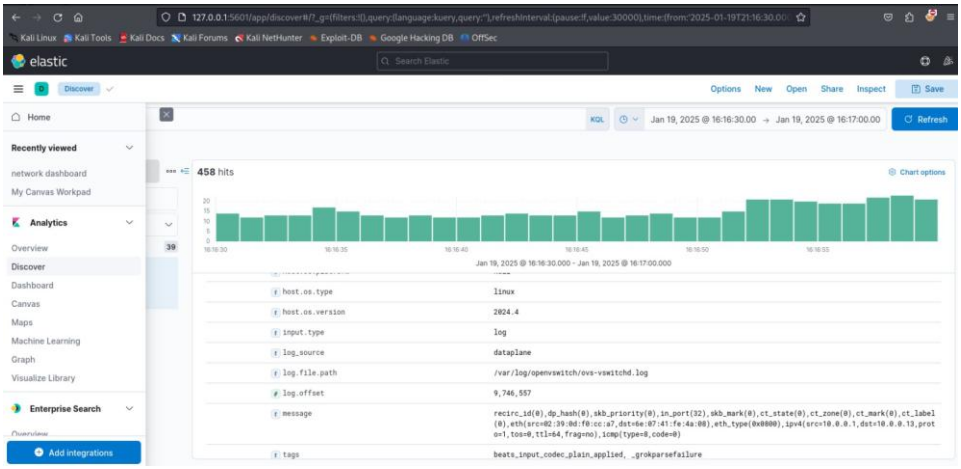


Figure 3. Kibana Discover View: Indexed and Parsed SDN Logs for Investigation

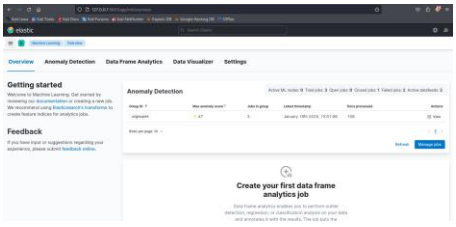


Figure 4. Elastic ML Anomaly Detection: Job Configuration and Detected Anomalies

supports investigation by allowing filtering and searching for suspicious fields (IP, protocol, controller messages, and error codes) and by correlating events over time.

Figure 4 shows the configuration of an Elastic ML anomaly detection job and its outputs. It highlights how the model learns “normal” behavior and flags deviations as anomalies, enabling alerts and reducing manual analysis.

5. Discussion

The experimental results demonstrate the effectiveness of the proposed integrated ELK and machine learning framework for real-time anomaly detection in an SDN environment. The system successfully centralized and processed logs from both control and data planes, provided intuitive visualizations of network activity, and quantitatively identified anomalies with severity classifications.

However, certain limitations should be acknowledged. The evaluation was conducted in a simulated Mininet environment. While useful for controlled experimentation, simulations may not fully capture the complexities, scale, and diverse traffic patterns of real-world production networks.

The anomaly detection mechanism, though effective for the simulated DDoS-like patterns (volumetric anomalies), was not extensively tested against a broader range of sophisticated SDN-specific attacks. The experience with the macof attack highlighted that certain detection tasks or more complex ML models might impose significant computational demands, underscoring a practical trade-off between detection comprehensiveness and resource consumption, especially for real-time systems. ELK stack deployments, while scalable, can also introduce management and cost complexities at larger scales.

6. Conclusion and Future Work

This paper presented the design, implementation, and evaluation of an integrated framework utilizing the ELK stack and machine learning for real-time anomaly detection in Software-Defined Networks. The system demonstrated its capability to effectively collect, process, and analyze logs from both control and data planes, providing valuable network visibility and successfully identifying anomalous activities indicative of DDoS-like attacks in a simulated environment. The classification of anomalies by severity further enhances the system's utility for security operations.

Future work will focus on addressing the identified limitations and extending the system's capabilities. Key directions include:

- **Scalability Testing**: Evaluating the system's performance, resource utilization, and detection accuracy in larger, more complex SDN testbeds with significantly higher traffic volumes and a greater number of network elements.
- **Advanced ML Models**: Incorporating more sophisticated machine learning and deep learning algorithms (e.g., Recurrent Neural Networks, Transformers) to improve the detection of subtle, zero-day, or evolving threats, and to reduce false positives.
- **Broader Attack Scenarios**: Expanding the detection capabilities to cover a wider range of attacks specifically targeting SDN architectures, such as control plane saturation attacks, malicious application deployments, or topology poisoning attacks.
- **Real-World Deployment and Evaluation**: Testing and validating the framework in a physical SDN testbed or a carefully controlled segment of a production network to assess its efficacy under realistic operational conditions.
- **Automated Response Mechanisms**: Investigating the integration of automated response capabilities, where detected high-severity anomalies could trigger predefined actions via the OpenDaylight controller, such as dynamic flow rule modifications to isolate malicious hosts or reroute traffic.

By pursuing these avenues, the aim is to develop a more robust, scalable, and intelligent security monitoring solution for dynamic SDN environments.

References

[1] Project Report: SDN Real-Time Anomaly Detection Using ELK Stack for Efficient Log Analysis and Visualization. (Internal Document Summary based on ProjectReportTex.pdf)

[2] M. A. Lodhi, S. M. Raza, M. A. R. Chaudhry, M. R. J. R.-C. S. Reviews, and undefined 2022, "Assessment and evaluation of security mechanisms in software defined network (SDN): A review," Journal of Network and Computer Applications, vol. 165, p. 102703, 2020.

[3] J. Su, H. Zhang, Z. Ling, Q. Xu, and X. S. S. N. Fu, "A Survey on Security-Aware Measurement in SDN," IEEE Access, vol. 6, pp. 26736-26754, 2018.

[4] M. T. Islam, M. H. H. S. M. R. K. M. M. A. S. S. N. S. U. M. M. Z. Hasan, "Integrating Software-Defined Networking (SDN) for Enhanced IoT Security: A Comprehensive Review of Mechanisms, Frameworks, and Future Directions," IoT, vol. 5, no. 1, pp. 338-378, 2024.

[5] "Is Elastic Stack (ELK) The Best SIEM Tool?," BitLyft, Sep. 13, 2023. [Online]. Available: https://www.bitlyft.com/resources/is-elastic-stack-elk-the-best-siem-tool

[6] "Elastic Stack: (ELK) Elasticsearch, Kibana & Logstash," Elastic. [Online]. Available: https://www.elastic.co/elastic-stack

[7] "Is the ELK Stack a SIEM?," Lumifi Cyber, Accessed 2024. [Online]. Available: https://www.lumificyber.com/blog/is-the-elk-stack-a-siem/

[8] A. H. Ali, S. A. Hussien, H. T. H. S. A.-A. J. o. E. C. S. Al-Dulaimy, and undefined 2024, "A Survey on Network Security Monitoring Tools and Functionalities," Mesopotamian Journal of CyberSecurity, vol. 2024, pp. 28-35, 2024.

[9] S. Agarwal, S. K. S. S. C. S. S. C. S. K. Singh, "Traffic Monitoring in Software Defined Networks using Opendaylight controller," Nokia Bell Labs, 2015.

[10] "OpenDaylight Project," OpenDaylight Official Website. [Online]. Available: https://www.opendaylight.org/

[11] "ELK Stack Pros and Cons: Is It Right for Your Log Management Strategy?," ChaosSearch, Accessed 2024. [Online]. Available: https://www.chaossearch.io/blog/elk-stack-pros-and-cons

[12] K. T. and P. R., "Simulation of SDN in mininet and detection of DDoS attack using machine learning," Bulletin of Electrical Engineering and Informatics, vol. 11, no. 5, pp. 2726-2735, Oct. 2022.

[13] A. Ananda and Y. Suarghana, "Systematic Review of Machine Learning-Based DDoS Detection in SDN Networks: A PRISMA Approach," ABEC Indonesia, vol. 8, no. 1, pp. 1-10, 2024.

[14] S. Jadhav and A. Kulkarni, "A Survey on Deep Learning Techniques for Anomaly Detection in Edge Computing," International Journal of Advanced Computer Science and Applications, vol. 12, no. 5, 2021.

[15] Al-Issa, "Deep Learning for Anomaly Detection: A Comprehensive Survey and Future Perspectives," arXiv:2503.13195 [cs.LG], Mar. 2025.

Declarations

Ethical Approval

We confirm that all procedures involving human participants were conducted in accordance with ethical standards. Ethical approval was obtained from the relevant ethics committee, and all participants provided informed consent prior to participation. Participants were fully informed about the study's objectives and their right to withdraw from the study at any time without any consequences.

Consent to Participate

All participants were informed of their right to anonymity and confidentiality. Their participation in this study was entirely voluntary, and they could withdraw at any time without any consequences.

Consent to Publish

The authors confirm that they have obtained consent from the participants to publish the results of this study. The published data do not allow for the identification of individual participants.

Data Availability

The datasets generated and/or analysed during the current study are included in this published article.