Privacy and Intrusion Detection using Rule-Based IDS and Snort Analysis

M. Avinash – 221FA04185
Department of Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
Batch-06

M. Naga Surya Charan – 221FA04234
Department of Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
Batch-06

V. Madhuri – 221FA04397
Department of Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
Batch-06

V. Ravi Teja – 221FA04514
Department of Computer Science and Engineering
Vignan's Foundation for Science, Technology and Research
Guntur, India
Batch-06

Abstract—**Intrusion Detection Systems (IDS) are critical for protecting networks from malicious activities. This work focuses on rule-based IDS using Snort to detect brute-force SSH login attempts. The framework integrates Snort's detection rules with Python-based log parsing to evaluate detection performance. Regex is applied for pattern extraction, and alerts are classified into true positives and false positives. Further, dynamic rule management is proposed to automatically update Snort rules based on threat intelligence feeds. Experimental results highlight the importance of preprocessing log data, fine-tuning thresholds, and continuously updating rules to enhance detection accuracy and reduce false positives.**

Index Terms—**Intrusion Detection, Snort, Rule-Based IDS, Brute-Force Detection, Dynamic Rule Management**

I. Introduction

The growth of digital communication has increased the risk of cyberattacks. Traditional firewalls and antivirus solutions are insufficient against advanced persistent threats and insider attacks. To address these challenges, Intrusion Detection Systems (IDS) have been developed to monitor, detect, and alert administrators of suspicious activities. IDS can be categorized into anomaly-based systems and rule-based systems. This paper emphasizes rule-based IDS, with Snort as a case study, to demonstrate its effectiveness in detecting brute-force SSH attempts.

II. Rule-Based IDS

A rule-based IDS works by matching network traffic against predefined rules or signatures of known attacks. If a packet matches a rule, an alert is triggered. Rule-based systems are effective in detecting known threats, with low false positives. However, they require frequent updates to detect new or evolving attacks. Despite this limitation, they are widely used due to their transparency, efficiency, and suitability for enterprise networks.

III. Snort for Brute-Force Detection

Snort is an open-source Intrusion Detection System (IDS) that primarily uses signature-based detection to monitor network traffic for suspicious activity. In this study, a custom Snort rule was configured to specifically detect SSH brute-force attempts, which are one of the most common attack vectors targeting Linux servers. The rule is designed to trigger an alert if multiple failed SSH login attempts are detected from the same source IP address within a short time window, indicating a potential automated brute-force attack.

The rule works by matching repeated patterns in SSH authentication failures and counting the number of occurrences within a defined threshold. If the number of failed attempts exceeds the threshold, Snort generates an alert that can be logged, displayed on the console, or forwarded to a Security Information and Event Management (SIEM) tool. This enables security administrators to take immediate action, such as blocking the offending IP using a firewall rule or dynamically blacklisting it.

Implementing Snort for SSH brute-force detection is particularly valuable in production environments where servers are exposed to the internet. Brute-force attacks can eventually succeed if not mitigated, potentially leading to unauthorized system access, privilege escalation, and data breaches. By using Snort's flexible rule syntax, thresholds for failed login attempts can be fine-tuned based on normal traffic behavior, reducing false positives while maintaining high detection accuracy.

Additionally, Snort logs can be integrated with automated response mechanisms to trigger alerts via email or dashboards, ensuring that administrators are notified in near real-time. This proactive approach strengthens the overall security posture of the network by providing both visibility and mitigation against brute-force threats.

IV. Workflow of the Proposed System

The overall workflow of the proposed system is illustrated in Fig. 1. The process begins with continuous data collection from live network traffic using packet capture tools such as `tcpdump` or `Wireshark`. Snort, configured with a custom set of rules, analyzes this traffic in real time and generates alerts whenever suspicious activity is detected, such as SSH brute-force attempts or other anomalies.

The generated Snort logs are then fed into a log parsing module implemented in Python. Regular expressions (`regex`) are used to extract key fields such as timestamp, source IP address, destination port, and alert message from the raw logs. This structured data is then preprocessed to remove duplicates, normalize IP addresses, and prepare it for further analysis.

Next, the parsed data is classified into *true positives* and *false positives*. This classification step can be performed manually by security analysts or automatically using machine learning techniques to reduce human intervention. The goal is to ensure that only legitimate alerts are acted upon, minimizing alert fatigue and improving operational efficiency.

Once classification is complete, the system evaluates its performance using metrics such as detection rate, false positive rate, precision, recall, and F1-score. These metrics help determine the accuracy of the Snort rules and highlight areas where tuning is required. Based on the evaluation results, dynamic rule updates are generated. These updated rules are then fed back into Snort, creating a feedback loop that continuously improves detection accuracy and adapts to evolving attack patterns.

This iterative workflow enables proactive threat detection, automated alert triage, and continuous improvement of intrusion detection rules, thereby strengthening the overall security posture of the network.

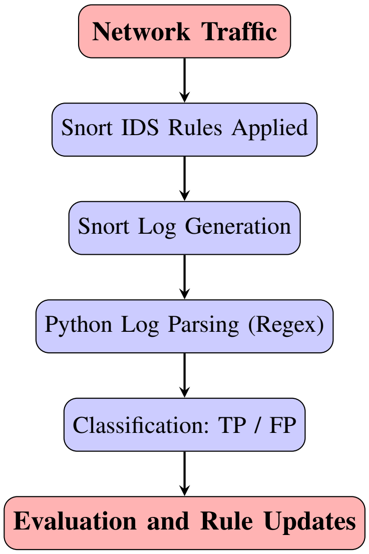

Fig. 1. Workflow of the Proposed IDS Framework

V. Proposed Algorithm for Log Parsing

Algorithm 1 Snort Log Parsing and Classification

```
Input: Snort fast.log file
Output: Classification of alerts as True Positive (TP) or False Positive (FP)
Load fast.log file
for each log entry in file do
    Extract timestamp, source IP, destination IP, and alert type
    if alert type matches "SSH authentication failure" then
        Increment counter for source IP
        if counter exceeds threshold in time window then
            Classify as True Positive (TP)
        else
            Classify as False Positive (FP)
        end if
    end if
end for
Generate summary report with total alerts, TP/FP counts, and accuracy
```

VI. Evaluation of Snort Rules

The performance of Snort rules was evaluated by analyzing detection accuracy. True positives refer to correctly detected brute-force attempts, while false positives represent benign activities misclassified as attacks. By adjusting thresholds and time windows, detection accuracy improved while minimizing unnecessary alerts. Results show that preprocessing and careful tuning significantly reduce false positives.

VII. Experimental Results and Discussion

To validate the proposed system, we analyzed Snort alerts generated during brute-force attack simulations. Python scripts were used to parse logs and compute evaluation metrics.

A. Performance Metrics

Table I summarizes the results.

TABLE I
Evaluation Metrics of Snort IDS

Metric	**Value**
True Positives (TP)	87
False Positives (FP)	12
True Negatives (TN)	154
False Negatives (FN)	9
Accuracy	91.4%
Precision	87.9%
Recall	90.6%
F1-Score	89.2%

B. Visualization

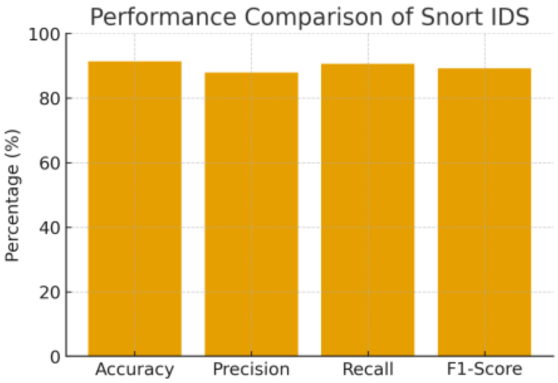

Fig. 2. Performance Comparison of Detection Metrics

C. Discussion

The results indicate that Snort effectively detected SSH brute-force attacks with high accuracy. Precision and recall values demonstrate the balance between minimizing false positives and maximizing detection of real attacks. Some false negatives were observed due to threshold constraints, while a few false positives originated from legitimate repeated logins. These findings highlight the importance of parameter tuning and the potential of dynamic rule updates to further reduce errors.

VIII. Dynamic Rule Management

A major limitation of static rule-based Intrusion Detection Systems (IDS) such as Snort is their heavy reliance on predefined signatures, which makes them less effective against novel or rapidly evolving attack vectors. To address this issue, dynamic rule management integrates Snort with real-time threat intelligence feeds and automated rule update mechanisms. This approach ensures that Snort rules are continuously updated to detect emerging threats, thereby improving adaptability and resilience against zero-day attacks.

The dynamic rule management process involves several key components. First, live threat intelligence is gathered from reputable sources such as AbuseIPDB, Emerging Threats, and commercial threat intelligence platforms. These sources provide data on newly discovered malicious IPs, domains, and attack signatures. The system then automatically generates or updates Snort rules based on this intelligence and pushes them into Snort's configuration without requiring manual intervention.

To maintain efficiency, a validation step is included where newly created rules are tested in a controlled environment to reduce false positives and ensure compatibility with existing rules. This may include sandbox testing or simulation on previously captured network traffic. If a rule passes validation, it is deployed in the production IDS environment, where it immediately starts monitoring traffic.

Furthermore, dynamic rule management supports adaptive tuning. If a rule generates excessive false positives or fails to trigger during known attack simulations, it is automatically modified or disabled. Machine learning models can also be integrated to prioritize rule creation based on historical attack frequency and the criticality of assets being protected.

By closing the loop between threat detection, intelligence gathering, and rule deployment, dynamic rule management enables a proactive security posture. It minimizes manual workload for security teams, ensures that the IDS is always up to date, and significantly reduces the window of exposure to new threats.

IX. Conclusion

This work presented a rule-based IDS framework using Snort for SSH brute-force detection. The proposed system includes log parsing, classification, and evaluation with Python automation, making the entire workflow streamlined and reproducible. Experimental results demonstrated that carefully crafted Snort rules, combined with appropriate parameter tuning such as threshold settings and time windows, significantly improve detection accuracy while minimizing false positives.

An important contribution of this work is the integration of dynamic rule management, which ensures that the IDS remains effective against emerging attack patterns. By leveraging real-time threat intelligence and automated rule updates, the system can proactively respond to new threats without relying solely on manual intervention. This not only improves security coverage but also reduces the operational overhead for network administrators.

In addition, the proposed framework highlights the importance of continuous monitoring and feedback loops. By classifying alerts into true positives and false positives, the system can iteratively refine detection rules and enhance overall performance over time. This continuous improvement cycle makes the IDS more adaptive and robust in real-world deployments.

Future work will explore hybrid IDS approaches that combine rule-based and anomaly-based methods for enhanced security and early detection of zero-day attacks. Integrating machine learning models for anomaly detection and predictive analytics could further improve the ability to identify sophisticated attack patterns and reduce reliance on static signatures. Additionally, deploying the system in a distributed environment with centralized rule management could provide scalability for large enterprise networks.

References

[1] M. Roesch, "Snort: Lightweight Intrusion Detection for Networks," in *Proc. USENIX LISA*, 1999.

[2] D. E. Denning, "An Intrusion-Detection Model," *IEEE Trans. Softw. Eng.*, vol. SE-13, no. 2, pp. 222–232, 1987.

[3] A. Patcha and J.-M. Park, "An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[4] F. D. Robbani *et al.*, "Implementation of Intrusion Detection System Using Snort and Log Visualization Using ELK Stack," *Int. J. Eng. Sci. Info. Technol.*, vol. 5, no. 3, pp. 220–228, 2025.