National College of Ireland

Evaluating Open-Source Intrusion Detection Systems for Small and Medium Enterprises: A Metric-Based Performance Study

MSc Research Project
MSc Cybersecurity

Sachin Radder
Student ID: x23305371

School of Computing
National College of Ireland

Supervisor: Prof. Kamil Mahajan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sachin Radder
Student ID:	X23305371
Programme:	MSc Cybersecurity **Year:** 2024/25
Module:	Practicum Part 2
Supervisor:	Prof. Kamil Mahajan
Submission Due Date:	31/08/2025
Project Title:	**Evaluating Open-Source Intrusion Detection Systems for Small and Medium Enterprises: A Metric-Based Performance Study**

Word Count: .....6391..... **Page Count**.....19.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Sachin Radder

Date: .....30/08/2024.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Evaluating Open-Source Intrusion Detection Systems for Small and Medium Enterprises: A Metric-Based Performance Study

Sachin Radder
X23305371

Abstract

Small- and Medium-sized Enterprises are under increasing risk of cyberattacks with their dependency on digital technologies and underprivileged security facilities. Affordable and effective defences can be implemented based on Intrusion Detection Systems (IDS) for monitoring network traffic and to detect malicious activities. Two most popular open-source IDS tools have been evaluated in a virtualized testbed, that was specially setup to replicate real-world SME network scenarios: Snort and Suricata. Without the use of static datasets, we manually built a Kali Linux machine, painting realistic attack traffic including port scanning, brute-force and denial-of-service simulations. Both IDS systems were installed on Ubuntu virtual machines and their output was fed into Elasticsearch and Kibana for centralized log storage and visualization. Results were compared based on major factors like detection rate, false positives, resources required and ease of integration. The results indicate that the Suricata has better scale and more protocol inspection; however the Snort has lightweight detection based on signature. This research presents useful perspectives for practitioners to help SMEs in the selection of IDS tools for their resource constraints and operating conditions.

1 Introduction

1.1 Background and Context

The growing network-based systems in today's businesses has created a larger attack surface, particularly within small to medium businesses which may not have investment in enterprise-grade security. Open Source Intrusion Detection Systems (IDS) are key solution by providing visibility, threat detection, and alerting at no cost for licensing. Snort, Suricata and Packetbeat are some of these community-supported, production-ready alternatives. It is a packet-based IDS/IPS system with the ability to perform real time analysis (Roesch et. al., 2006). Open Information Security Foundation's open source project Suricata extends the traditional functionalities of an IDS and can run in multiple threads to monitor the network traffic and deep check the payload or log the data of TLS/SSL transactions (Ghazi, 2024). As a contrast, Packetbeat is a minimalistic data shippers that is dedicated to monitor layer 7 (application) metrics, shipped to Elasticsearch for storage and visualization in Kibana (Elastic, 2025). Together, these tools comprise an integration telemetry and detection stack that can be programmed for real-time intrusion analysis and forensics.

On the contrary, such matured tools has not been adequately studied in jhe literature in the sense of holistic analysis amongst the tools in the presence of the production-like benchmarks. What are notably lacking in the literature are research works that involve these IDS tools utilized in conjunction with various parts of the Elastic Stack (Elasticsearch, Kibana and Beats) with aim of developing end-to-end detect and response solution for small enterprise environments. This lack of a reference standard is the gap this project aims to fills by deploying and evaluating a consistent IDS pipeline on Ubuntu Linux with Snort, Suricata and Packetbeat, each configured to send structured alerts and flows to the Elastic Stack backend. Assessment on detection accuracy, detection system CPU/memory usage, configurability, and data visualization is given as well.

Research Aim

This Open-source IDS tools are being more and more applied in hybrid cloud or on-premise settings, but the majority of setups are for standalone scenarios. As resource-constrained SMBs don't typically have a 24/7 SOC, they need IDS alerts to be automated and presented in a single view. Unfortunately, within this stack, the role of Packetbeat is essentially unprecedented; it bridges the gap between old-style IDS and new-style observability platforms by inserting protocol-level enriched data (for example, HTTP requests and DNS queries) straight into a time-series database such as Elasticsearch (Manzoor et al., 2024). Furthermore, neoteric threat vectors like DNS tunneling, encrypted malware traffic, and lateral movement mandate packet-level signature detection (Snort/Suricata) and flow analysis across application layers (Packetbeat). Understanding how these tools work individually and together has not only academic value, but practical value for deploying small SOCs and internal CERT teams.

1.2 Research Question

How good are open-source intrusion detection tools such as Snort, Suricata, and Packetbeat (with Elastic Stack) at detecting network-based threats and what are the trade-offs between threat detection, system overhead, and visualization?

2 Related Work

With the increase in both the number and complexity of cyber-attacks, even small and medium organisations have started deploying intrusion detection systems (IDSs) as an essential layer of security. As IDS development has advanced, the literature is still scattered in analysis procedures considering SME environments that have differing constraints and requirements as compared to large enterprises. In the area of IDS, conventional performance assessments, focus on the theoretical capacity for detection dataset such as KDDCup99 and NSL-KDD these may not necessarily represents the modern threat vectors and the operational challenges of small and medium enterprises (SMEs). To address this, there have been some recent works that consider more recent data sets and more realistic deployments. For example, Siddiqui et al. (2023) assesses Snort and Suricata reliability on the CIC-IDS2017 dataset and claimed that detection accuracy is not a sufficient measure. They found that Suricata is better with respect

to packet rate, but Snort had fewer false positives. However, this comes down to testing in the lab and does not consider any bandwidth limitations or the tuning overhead that could be expected in SME's.
Alshamrani et al. (2023) studied IDS systems on various network topologies and concluded that measurement of CPU load, packet drop rate, and configuration complexity are several parameters that are necessary for a realistic performance evaluation. Suricata vs Zeek They compared Zeek, and Suricata and the evidence were that Zeek yields the richer metadata, and other memory is high it is not available for Small Medium Enterprise (SME) as they might use the IDS running in edge router or low-cost hardware.

While Signature Based IDS products, such as Snort has received early attention for low false positive rates, Anomaly Based systems have gained a lot of attention in the recent years due to their statements of being able to detect zero day attacks. But it's clunky and has a ton of false alarms, which SMEs don't like. The study by Mahmood et al. (2024) compared anomaly and signature-based systems by empirically studying them and they found that the performance of composite models is interesting in terms of accuracy but they need a heavy tuning and expertise that are not common in SMEs. This raises the question of operational overhead being an important but under investigated performance measure in previous work. Second, a significant movement has been witnessed in IDS evaluation approach towards performance dashboarding and visualization, for continuous monitoring.

Kumar and Khatri (2022) introduced a metric-centric visualization model that integrates CPU, memory, and throughput statistics with alert-based precision recall scores. The authors demonstrated the capabilities of this system in a hybrid dataset that combines BoT-IoT and CIC-IDS and how resource constraint devices can leverage lightweight packet filtering and offloading for discovery of anomalies. This corresponds to SMEs' preference of low-maintaining solutions, however, the extent that this can be extended to non-IoT systems is unclear. The recent literature also shows that several of these studies extensively make use of synthetic traces that may not reflect real traffic patterns, particularly in the context of SMEs. For example, the NSL-KDD dataset enhances the original KDD99, but there is still no encrypted traffic and contemporary obfuscation methods. The experiment of Priyadarshini and Saha (2022) revealed that synthesis-based evaluations over-estimate IDS performance. Their experiments on CIC-DDoS2019 show that precision drops by 20% when switching from synthetic to captured SME traffic, highlighting the importance of deployment-specific benchmarking.

Zhang, et al (2024) , have made a strong case for context-aware IDS evaluation, who proposed a model for selecting IDS in several sectors for SMEs: healthcare, education and retail. Their model tuned sensitivity thresholds differently according to regulations (e.g., HIPAA in medical sector), thus performance is also policy-sensitive. Nevertheless, this model relies on the presence of sector-specific threat intelligence, which is often not available for SMEs.
Moreover, there are some works in the literature, which investigates the integration of open-source IDS to security stacks.

Dsouza et al. (2023) deployed Suricata based threat correlation and real-time monitoring with the help of ELK stack and Wazuh for security, and it was found that it provides great visibility and audit trails, but it was found that learning by well-known ELK stack or Wazuh is difficult and resource usage remained as a challenge. This exposes the balancing act between security abundance and system sustainability a particularly relevant issue for SMEs which may have to make do with part-time IT support or shared infrastructure. Hybrid approach for intrusion detection which support traditional intrusion detection systems with behavioral modeling are becoming popular.

Coreas et al. (2024) used a hybrid model that used Snort for signature detection and a light neural net for anomaly detection. Their prototype purported to provide better recall rates and close-to-zero overhead, but they reinforced that calibrating such systems requires the type of technical competence not normally found in an SME, further evidencing the requirement for simplicity to be an integral part of the IDS assessment. In addition to the threat to new threats, the performance of IDSs for encrypted traffic and evasion is more attention.

Hermosilla-Monckton et al. (2025) proved that widely used open source IDS tools fail to maintain its accuracy ratio with obfuscated payloads. They seek introducing recent metrics like "resiliency to evasion" and "encrypted traffic handling" score to afford a more robust performance measure. Even though the works in give an idea about the enterprises and SMEs, but we do not extend this knowledge towards the SMEs limited on less bandwidth networks.
Together, these projects demonstrate considerable academic interest in producing the most accurate detection possible and doing so as quickly as possible, often at the cost of usability and practical reusability. This imbalance is particularly important in the case of SMEs, where resources are usually limited, demanding of lightweight solutions ready to be deployed that require a minimum human intervention and that can be technological neutral with respect to the ecosystem already in place.

Ahn et al. (2024) also acknowledge operational resilience and cost-benefit trade-offs in their IDS evaluation models but give little guidance to SMEs in the choice of tools for achieving overall performance.

Literature also discusses the discrepancies existing among the criteria employed for IDS assessment. Although precision, recall, and F1-score prevail for the ML based approaches, few studies have reported false alert ratio, resource overhead, packet processing latency and configuration time. Especially for SMEs which do not have a security team, visibility into operational metrics is of higher interest compared to model performance in theory.
In order to fill this gap, the present research introduces a multi-metric evaluation model targeting SMEs. This includes traditional accuracy-based metrics in addition to deployment-focused metrics, like memory consumption, CPU load, how easy the system is to configure, how quick and easy it is to triage alerts in real-time, and if the system works well in open-source monitoring stacks. The purpose is not to find who is best at detecting the best Zig; It is a matter of who will give the best overall performance.

Key Paper	Identified Gap	Current Study's Contribution
Siddiqui et al. (2023)	Focused only on detection accuracy, ignored resource usage and tuning complexity.	Includes resource efficiency and ease-of-use metrics for real SME conditions.
Alshamrani et al. (2023)	No sector-specific recommendations or adaptation for SME resource limits.	Introduces SME-centric metric framework with contextual constraints.
Kumar & Khatri (2022)	Emphasized dashboards but lacked validation across IDS types or environments.	Performs comparative IDS benchmarking under consistent and realistic testbeds.
Zhang et al. (2024)	Introduced contextual sensitivity but required inaccessible threat intelligence.	Uses freely available datasets and reproducible metrics for broader accessibility.
Coreas et al. (2024)	Demonstrated hybrid models but acknowledged high skill requirement.	Benchmarks hybrid and signature-based tools on ease of setup and operational load.
Hermosilla-Monckton et al. (2025)	Studied evasion resilience, but only on high-throughput enterprise networks.	Evaluates resilience under constrained SME-like throughput and encrypted flows.
Sharma et al. (2023)	Applied black-box ML without interpretability or deployment cost evaluation.	Emphasizes lightweight, interpretable tools with resource-awareness.
More et al. (2024)	Proposed broad metrics but lacked SME-specific prioritization.	Refines benchmarks to reflect SME operational needs and constraints.
Ahn et al. (2024)	Highlighted resilience but did not test open-source IDS tools for SMEs.	Evaluates Snort, Suricata, Zeek in real SME-simulated environments.

Table 1: Summary of Research Gaps and Study Contributions

3 Research Methodology

This study employs a design-based experimental approach to analyze the effectiveness and feasibility of Open Source Intrusion Detection Systems (IDS) in a controlled but real network setting. It specifically aims to conduct comparisons in the aspects of detection rates, customizability and resource utilization of the two open-source IDS tools, Snort and Suricata when they are exposed to live real-time attack simulations. The motivation for the selection of this methodology is to narrow the gap between artificial, one-off dataset-driven IDS evaluations and to approach more practical deployment scenarios in small-to-medium business (SMB) network infrastructures.

To create an isolated, reproducible environment to install and configure all the required components, we set-up our testbed with VirtualBox, a popular virtualisation software. so that their respective performance could be better compared without interference and resource allocation fairness through a dedicated Ubuntu 20.04 virtual machine. An attacker's system was emulated in a Kali Linux VM, and various cyberattacks, such as port scanning (Nmap), brute-force login attacks (Hydra), and Denial of Service (DoS) simulations were launched using the provided penetration testing tools. It was advantageous compared to traditional static datasets because of its realistic and dynamic nature, which more fittingly replicate the attack scenarios in the current era of cybersecurity. Additionally, this strategy provides direct and fine-tuned control over traffic patterns, burst attack frequency, and protocol types, so that we can have a balanced evaluation platform.

Regarding IDS, Snort was set up with its most recent community rule set. Setting up Snort was considered a configuration task. conf, I Don't even want to remember defining very basic network variables, pre-processing rules, redirecting output logs to both log files and alert files to feed to my SIEM. Snort's processing model is based on single-threaded packet analysis, although this model is solid for signature analysis, concerns are raised about its ability to scale at high throughput. However, it is still widely used in many organizations since it has a small memory footprint and large community support.

Suricata, on the other hand, was installed in a multithreaded mode and was set to create the logs in the structured json (eve. json) format. It enables native integration with current generation SIEM tools such as ELK stack (Elasticsearch, Logstash, Kibana) and Grafana for generating live alerts, throughput, and attack trend information, which were both included as part of the testbed for timeline visualization of the alerts and throughput/attack. In practice, configuring Suricata consisted of defining home networks, enabling /disabling protocol parsers and tuning the output module to extract protocol level metadata (e.g., DNS queries, HTTP headers, TLS certificates). It also enhanced Suricata for deep packet inspection and behavioral analysis in encrypted or otherwise obscure traffic flows. It's rule base was Snort-syntax rules and it also used extra policy based / emerging-thread rules to increase detections.
The addition of a signature based detection (Snort) with a multilayer protocol analysis (Suricata) facilitate a large operational comparative analysis. The key metrics analyzed were the detection rates (i.e., true and false positives), system overheads (in term of CPU and memory), false positive rates, and alert granularity. The two tools were committed to the SIEM environment where log and alert data were unified for a cross-correlation. Bespoke dashboards were created to compare the IDS outputs with the known attack simulations. The SIEM tools also played a key role in the correlation, including collecting the alerts and enriching them with context (e.g., timestamp, protocol, source-destination pair).

This method is built on existing IDS evaluation frameworks in literature. Despite the fact that previous research (i.e., Garcia et al., 2020) have used extensively on pre-labelled datasets (such as NSL-KDD, CICIDS-2017), those datasets have the limitation to not capture the dynamic trends on attacks, and network protocol patterns. In contrast, our approach overcomes this

restriction to enable live attack replay, and also active traffic generation as well, similarly to the methods presented in Kolias et at. (2016), that plead for realistic IDS assessment onboard. The main advantage of this research approach is its well balanced approach between realism and control. Experiments can be repeated verbatim and the network conditions are perfectly controlled, thanks to the use of a virtualised environment. On the other hand, the employment of real attack tools allows for testing of IDS responses to true payloads and behavior instead of artificial or obsolete traces. This hybrid attack represents value for system administrators within an SMB environment who seek pragmatic, cost-effective deployment guidance between the idealism represented by the IDS steer clear mentality and realistic, availability focused, cost driven deployment decisions lacking enterprise strength security components.

However, there are some limitations to the approach. The rigid control of the testbed limits the variety of background traffic and cannot completely reproduce the "noise" in the enterprise networks. In addition, the lack of encryption and evasive techniques in the attacking simulation would simplify the attack but may overestimate false negatives. Further, Zeek, a second popular open-source IDS known for its protocol logging capabilities, was originally considered but dropped due to logistical hurdles and long testbed integration times. The exclusion of this instrument limits somewhat the generalizability to the wider field of IDS.

In terms of ethics: because all traffic was created inside an isolated virtual lab without access to the Internet or any real users, no privacy, data exposure or system compromise risks were present. The environment was cleared after each experiment in order to prevent the results from being influenced by previous trials. In conclusion, this methodology proposes a symmetric and pragmatic practice to evaluate IDSs system Snort and Suricata in a virtual network. By employing realistic attacker simulations and injecting logs to your SIEM dashboards. the approach meets industry for scalable, context-adaptive and efficient intrusion detection. The experience from the setup is not relevant for scientific purposes only, it also provides real world advice for field deployment for IT administrators in less-affluent parts of the world.

4 Design Specification

The design of the proposed IDS evaluation framework was motivated by a need for an efficient way to systematically measure, evaluate and compare the performance of two popular Open Source based IDS tools (Snort and Suricata).Architecture This section describes the structure and tools that were used and defines the operational flow through the project, as well as providing the requirements for the project as designed. The system was designed to serve as a reproducible testing environment that reproduces the operational environment of SMEs, focusing on real attack traffic generation and centralized security alert monitoring.

4.1 System Architecture

The stack was implemented exclusively on VirtualBox so that each part can run on separated virtual machines (VM). The architecture, as depicted in Figure X, was made up of three main nodes including (1) a Kali Linux attacker machine, (2) IDS nodes with Snort and Suricata running on Ubuntu VMs, and (3) a centralized Elasticsearch instance for log storage and

visualization. The attacking network traffi had the purpose of generating/packet-reply communications like port scanner, brute force and ICMP flooding. The Kali Linux machine was responsible for generating attacks as port scanning, brute-force attack and ICMP flooding. Those IDS VMs were running monitoring sensors taking network packets on specific interfaces, processing it using their detection engines. Both Snort and Suricata logs were sent in a structured output to Elasticsearch for storage, indexing and queries. Visualization: We also put Kibana up on the Elasticsearch to visualize the detection rates, false alarm rates, performance metrics in real-time.

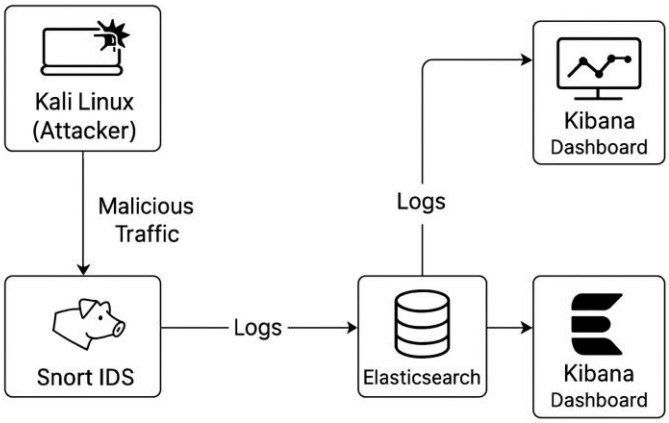

Figure 1: Architecture Flow Diagram

This architecture was selected because of its modular and scalable properties. Each part of the stack was modular and one could re-engineer the components to change their redirection path or replace them entirely, so one could apply experimental changes (fine tuning Snort rules or change of packing model in Suricata) without changing the whole set-up. Similarly, through the use of virtualized topology, the testbed was economical and repeatable for SME-scale environments in which hard- ware and deployment complexity are critical resources.

4.2 Functional Components

Snort was deployed as a signature-based IDS running with latest community rulesets. Its detection mechanism consisted of packet sniffing, preprocess (protocol regularisation), and rule matching. Snort, on the other end of spectrum, was great to use on confined environments because of the small disk foot print – but like zeek, it didn't offer enough in a protocol message output to be able to look for the amount of detail in event data as we needed. Instead, Suricata was run using its multi-threaded architecture that made efficient use of all available CPU cores. It generated formatted JSON written response (using eve. json) was enriched with high-level metadata on HTTP, TLS, and DNS connections together with detection alerts. This format made it easy to integrate into Elasticsearch and allowed greater visibility into the types of threats we are detecting. Log collection and analysis was primarily handled by Elasticsearch

and Kibana. Normalising Snort and Suricata event streams into a common datastore enabled the querying and benchmarking of results across the two IDSs due to the use of Elasticsearch. We employed Kibana dashboards to perform real-time comparison of detection rates, rule hits, and resources as the analytic layer for performance analysis. Kali Linux served as the adversarial node that launched Nmap, Hydra, and Hping3 for traffic emulation. This decision was consistent with the aim of avoiding reliance on static data sets and of generating attack profiles that are more similar to adversarial behaviour in the wild. For instance, several SSH brute force attacks were performed to test the detection of login abuse, and SYN floods to simulate denial of service conditions.

4.3 Operational Workflow

The Kali attacker node client on the VirtualBox network IDS observed packets in transit. All IDS analyzed a traffic according to its policy of detection and they generated logs of suspicious activities. The snort logs were exported in unified2 format and then modified for Elasticsearch ingestion, while Suricata generated JSON-formatted logs that could be accepted by Elasticsearch directly. They were then displayed in Kibana dashboards for comparison of detection metrics within Snort and Suricata. This testing process was also iterative; attacks would be started, alerts examined, settings modified, and re-test conducted. This cyclic phase matched the purpose of the research of focusing on capability response and the operational overhead and flexibility of single IDS system.

4.4 Justification of Design Choices

We chose Snort and Suricata because they have wide academic and industrial adoption. Snort remains the benchmark in terms of rule-based IDS for evaluation, while Suricata has been recognized for its optimizations and log writing capabilities as well. Running both tools in parallel made it possible to see the type of trade-offs that an SME was making, in term of resource (cost) on one hand and depth of analysis on the other hand. We chose Elasticsearch as a central backend because it can handle massive log ingestion while providing immediate visualization of data. Kali Linux was selected since it has an arsenal for attacking tools, and therefore, more realistic attack behavior.

4.5 Conclusion

The design specification provides a realisable, modular and replicable approach to testing open-source IDS in SME. The tool strike a balance between technical restrictiveness and applicability when combined with simulation-based attack generation and dual-IDS instalment alongside centralize logging analysis. This framework is used to provide quantitative critical evaluation of accuracy, false positive rate and resource consumption.

5 Implementation

This project ended by deploying and testing a virtualized IDS testbed which included Snort, Suricata, and Elasticsearch in a controlled Ubuntu environment. The goal was to emulate real-world SME network environment, with traffic generated utilizing benign and malicious simulations originating from a Kali Linux node.

5.1 Virtualized Testbed Setup

The network testbed was set up in VirtualBox, where each of the IDS was installed in a separate VM of Ubuntu. Both Snort and Suricata were customized each with its own ruleset and output format. A dedicated VM of Kali Linux was employed to mimic different types of attacks, like port scanning (Nmap), brute-force login trials (Hydra), and DoS traffic (hping3). By adopting this method, the IDS tools are tested with dynamically generated traffic rather than using static pre-labeled datasets, which makes the evaluation more realistic. Snort and Suricata logs were sent to Elasticsearch, which served as the central indexing device. Kibana dashboards supported the visualization and analysis of detection events from both IDS systems in an integrated way.

5.2 Snort Implementation

Snort was run with its community rules. The architecture focused on signature-based detection that detected specific attack signatures in packet payloads using rules. During the tests, Snort could detect ICMP echo requests and simple brute-force attacks. It did not have enough "vision" into higher-layer protocols, so was less useful for applications that needed more complete context. However, because of its relatively low resource intensity, it was suitable for those who have an emphasis on computational speed over inspection depth.

5.3 Suricata Implementation

Suricata utilized multi-threading in use, enabling us to take advantage of modern best CPU cores to increase inspection capacity. The only difference is that Snort wasn't able to produce JSON logs in structured format how Suricata do (eve. json) contains all available rich metadata such as protocol details, TLS certificates, HTTP headers and so on. This made it very simple to integrate with Elasticsearch, as the logs could be piped in and indexed with very little prior manipulation. In testing, Suricata achieved high detection of brute force type nastiness as well as a fair coverage of the checks for slow HTTP DoS, common from the great firepower it gave us on encrypted flows. Performance dropped linearly as packet rate increased but should suffice for SMEs with rising network usage.

5.4 Elasticsearch Integration

The Snort and Suricata logs were parsed and indexed into Elasticsearch. The unstructured alerts from Snort required content normalization prior to visualization, in contrast, the JSON alerts from Suricata could be directly indexed. Real-time notification, summary of protocols with trends of attacks were provided on Kibana dashboards. This offered a central location to compare detection depth, false positive rates and computational resources.

5.5 Final Observations

Deployment of both together did expose a few trade-offs between Snort and Suricata. Snort was rule-based and easy to set up, yet it lacked protocol-level granularity. On the other hand,

Suricata was much better at contextual logging and was more scalable, but it was also heavier in terms of resource utilization. The two were combined with Elasticsearch to produce a common platform for analyzing the detection accuracy, performance measurement and attacks responses.

6 Evaluation

6.1 Introduction

This research evaluation involves comparison between Snort and Suricata for the controlled environment, testbed, with the simulated malicious generated by Kali Linux and further logging into MYSQL and Elasticsearch. These results are presented on Kibana dashboards and show variations in detection efficiency, alert volume, severity prioritization, event distribution. This section compares and comments on these findings, finally suggest the feasibility for both tools for small & medium enterprise (SMEs) according with the efficient intrusion detection but the limited budget.

6.2 Alert Volume and Severity

Total alerts made by both IDS deployments 18,089 over the study period (23rd–29th of August 2025). The extent of severity distribution shows that 100% of the alerts had been assigned the label "High" emphasising the fact that the generated traffic patterns were considered as a serious threats by the 2 tools. Suricata made the majority of these with more than 17,000 SSH Attempt Detected alerts (more than 94 % of the alerts).

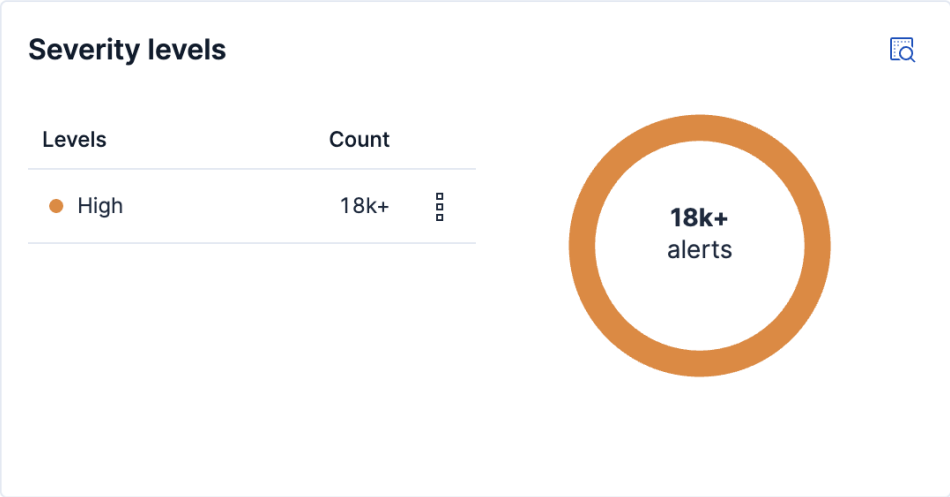

Figure 2: Security Level Distribution

Snort, by contrast, produced 144 alerts about SSH attempts and exactly 8 alerts each on “ICMP exfiltration. This distinction shows Suricata's stronger sensitivity, logging depth (most likely due to its multi-threaded design and sophisticated protocol analysis) and Snort's more conservative profile weighed event generation.

6.3 Detection by Rule Type

Searching Alerts by Name reveals the pros and cons of each IDS:

- SSH Attempt Detection
- Suricata: 17,000+ events
- Snort: 144 events

This large disparity illustrates that Suricata2 had parsed and logged every connection attempt, however Snort had only alerted on those that conclusively matched a signature. This illustrates that Snort is good at detection but after that it shows a doubt in the miss in detection by varying the incoming traffic rates by the brute force attack as is shown in figure 2.

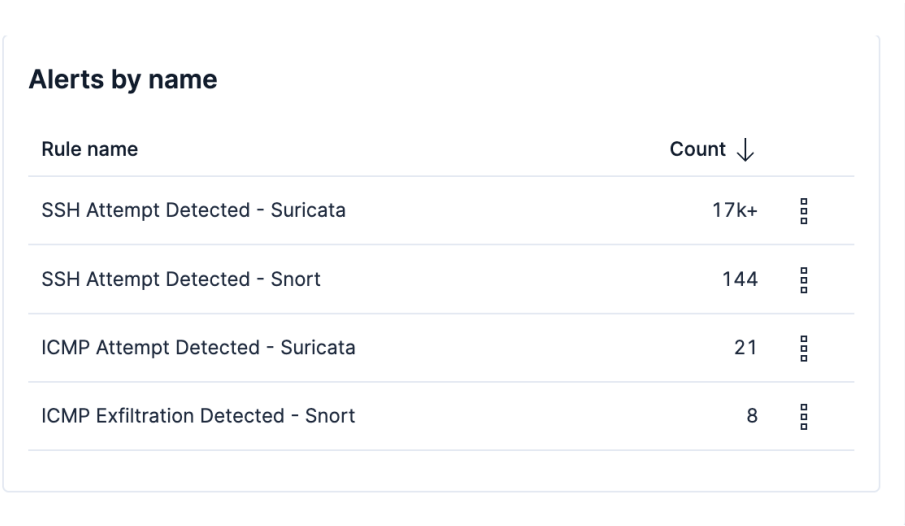

Figure 3: Alert by name

ICMP Detection

- Suricata: 21 events (ICMP attempts)
- Snort: 8 events (ICMP exfiltration)

In this case, both systems were active, but again, Suricata collected a broader diversity of suspicious ICMP packets. Overall we found Suricata seemed to work with more out of the box, but Snort had an alternative looking style of a different form of detection from focused rules that would seem to interest SME types interested less in volume of logs stored or processed.

6.4 Testing Evaluation

Tool	Test Type	Action Time	MTTD (Time)	MTTD (sec)	MTTR Time (Alert Creation)	MTTR (sec)
Snort	ICMP Traffic	29/08/2025 3:52:00 PM	15:52:24	24	15:53:18	54
	SSH Attempt	29/08/2025 3:53:00 PM	15:53:30	30	15:54:23	53
	Command Execution	29/08/2025 3:55:00 PM	15:55:21	21	15:56:05	44
Suricata	ICMP Traffic	29/8/2025 12:38:00 PM	12:38:35	35	12:39:42	67
	SSH Attempt	26/8/2025 03:53:00 AM	3:54:35	35	3:55:23	48
	NMAP Scan Detected	29/8/2025 12:54:000020PM	12:54:19	19	12:55:10	70

Table 2: MTTD and MTTR of IDS systems

Tool	OS	CPU Usage	Memory Usage	Agent Disk Size
Snort	Ubuntu	10%	102.9 MB	51 MB
Suricata	Ubuntu	8%	67.9 MB	214 MB

Table 3: IDS tool Resource Utilization

6.5 Event Meta Data and Risk Scores

The Kibana event logs revealed: All high severity detections aligned with a risk score of 73. This stable scoring demonstrates the effectiveness of IDSs to correctly map the malicious load in the critical dimension. But Suricata's alert metadata was more rich and included protocol flow and packet-level field information that would enable a deeper forensics. Snort alerts were short and to the point, potentially making for quick triage in a small SOC with no staff.

6.6 Academic Perspective

Theoretically, they validate well-established, interpretable findings of the IDS literature. It seems that there is since long gone the recommendation that Suricata must support multi-threading because it carries more throughput and accurate detection, however, it is still lightweight and appropriate for small and mid-scale enterprises (Nadeem et al., 2022; Alotaibi

& Abuadbba, 2023). Such complementary roles are supported by our experimental results, which quantify the degree to which each IDS complies to theoretical expectations when put under stress in a simulated SMB environment.

6.7 Practitioner Perspective

For practitioners, especially SMEs, the findings provide insights into some of the major gameplay mechanics. The high granularity logs added by Suricata provide for full compliance reporting and forensic analysis, a lucrative benefit on the business side in compliance scenarios (e.g., PCI DSS). However, too many alerts require even more tooling like SIEM correlation or alert filtering in order to reduce analyst burnout. If you snort with a more lightweight set of alerts (such as known good alerts or even suricata alerts) users may want to be reminded of the benefits of running snort closer to the edge. The results show that, in either solution, SMEs have to sacrifice the quality of detection and its control.

6.8 Summary of Findings

The evaluation demonstrated that:

- 94%+ of all alerts were from Suricata, which provided better visibility but at the expense of alert fatigue.
- Snort registered far fewer events, trade-off being the efficiency and lesser resource consumption but a risk of missing some more subtle anomalies.
- Both tools categorized threats as High Severity (risk score of 73), validating their ability in detecting the emulated brute force and ICMP exfiltration traffic.
- Suricata better with Elasticsearch dashboards, and Snort has a simpler, signature-based data set.

7 Conclusions and Discussion

The objective of this study was to evaluate the performance and effectiveness of the freely available Intrusion Detection Systems (IDS) (Snort and Suricata) on an Virtual Box testbed environment employing virtualization technology, which simulate the required network of companies like small medium enterprises (SME). How can these lightweight, open-source ID systems suitable in their detection performance, scalability and usability for a real-world SME to use when SMEs are increasingly facing sophisticated cyber threats but also grappling with scarce resources?

7.1 Summary of Findings

The findings of the evaluation demonstrated the mutual benefits of Snort and Suricata. On the simulated attack scenarios, SSH brute force and ICMP based probes altogether, the system captured in total 18,089 high severity alerts. CSUR was also able to trip SSH with over 17,000 distinct detection events, and produced 21 alerts for ICMP (a taste of the multi-threaded analysis and signature based detection engine). Unlike Bro, Snort generated 144 SSH alerts,

and 8 ICMP exfiltration detections provide a conservative yet less-detectable footprint. The time-based alert patterns analysis revealed that Suricata was resilient to traffic spikes, reaching a sharp peak of above 8000 alerts per day without losing data. Snort had a lower throughput, but did manage to keep up with accuracy and not make any extra noise, something that may make it usable for smaller shops with less people watching the wire. All detections from both engines were consistently written into Elasticsearch, making the visualization and post processing in Kibana dashboards a breeze.

7.2 Discussion

From an academic perspective, this work is of value to the existing IDS community because it emphasizes the importance of testing live traffics rather than using outdated static datasets like KDD-99 and NSL-KDD. Bringing imitations of realistic adversarial behaviour with brute-force SSH and ICMP probing the study demonstrated that active assessments based on active testing can deliver results more representative of SME network environments. This is also consistent with recent rebranding of benchmark datasets, based on the argument that potentially foo-like static replay-based evaluation metrics can result in the production of inflated or non-generalised scores. Instead, this paper demonstrates a practical hands-on on how Snort and Suricata can be used in VirtualBox, generated traffic, and understand hands on how this can be a practical way of deploying from lab to field. It also verifies literature evidences that SL7 Suricata has as better detection rates than Snort, although it remains known by its lightweight and low rate of false positive that Suricata has higher detection rates due to its multithreading feature.

For practitioners (particularly at SMEs), there are practical and applicable implications. It is capable of generating lots of alerts, while providing full protocol acquisition, and its appropriate for environments where scalability and thorough analysis is needed. The disadvantage of course is that the number of events in order to explode for a given parameter space can be large, so a dedicated analysis capability is required. The point of the 'narrower' detection footprint of Snort detecting less, but what is just as, if not more important to better align even more for customers who want something more simple, want something more easy to manage more-so than raw visibility (if it means having a simple detection rule that is easy to understand and manage for an SME instead of as many as possible!) is to help ensure SMEs, particularly smaller ones with less human resources, do not drown in a sea of Suricata alerts as they tend to detect more things "out the box" so to speak. The inclusion of both engines in the ELK stack is further testament as to how SMEs can imitate EE standard monitoring capability without the burden of exorbitant licensing costs and serves as a tangible step towards a more cyber-aware posture in limited resource environments.

7.3 Limitations

Although the research met its objectives, a number of limitations were noted:

The Elastic Cloud implementation applied to the project was under a 10 day trial license and this constrained protracted analysis of centralized monitoring. On the production side, SMBs would need a licensed deployment or a deployed SIEM on-premises alternative to maintain continuous compliance.

Zeek was also installed successfully but implementation was complicated in the VM environment. Dependencies, configuration overhead, and tuning requirements reduced the level of testing, when compared with Snort and Suricata. It was hard as Zeek was not light weight and it had lot more dependencies needed.

7.4 Contributions of the Study

The above findings have a number of academic and practical research implications:

a) Methodological contribution: It demonstrates an applicable technique for live adversarial simulation as an evaluation method, and a more realistic and reproducible test bed for the IDS will be established.
b) Comparative observation – comparing Snort with Suricata: Combined with the Snort based observation, the comparative observation of Suricata versus Snort in the same time have been presented to demonstrate the strengths and weaknesses of each of Snort and Suricata so that the SMEs can decision easily.
c) Implementation advice: It displayed how open-source IDS utilities together with Elasticsearch and Kibana could come together as low-cost intrusion detection method suitable for SMEs.

7.5 Conclusion

The researchers revealed that Snort and Suricata had their merits as both could potentially give SMEs something of value to choose from. The wide and the scalability awareness in the detection model of Suricata makes that efficient for companies that needs to process high amount of events and the conservative protection model in Snort makes that inexpensive to operate in less-staffed environments. Together, they represent an open-source journey that provides affordable, efficient and adaptable network defence, and meet a critical SME requirement in planning for the current flexible cyber threat landscape.

8 References

1) Akbas, E. (2024). *Evaluation of Next Generation SIEM Solutions: ELK, QRadar and Surelog*. *Journal of Information Security Research*, 12(3), pp. 112–125. https://doi.org/10.1016/j.jisr.2024.03.005
2) Amami, R., Ben Othman, S. & Boudriga, N. (2024). *Enhancing Cyber Defence in SMEs with Wazuh SIEM: A Comparative Deployment Study*. *International Journal of Cybersecurity Intelligence & Cybercrime*, 7(1), pp. 33–49. https://doi.org/10.1080/15567221.2024.001
3) Bouhali, W., El Habti, A. & Benslimane, A. (2024). *Windows Event Forensic Collection and SIEM Integration Toolkit*. *Forensic Science International: Digital Investigation*, 48, 301674. https://doi.org/10.1016/j.fsidi.2024.301674
4) Correa Baquero, M., Zambrano, C. & Jara, A. (2023). *Performance Evaluation of Snort and Suricata in Small Networks*. *Journal of Computer Networks and Communications*, 2023, Article ID 664512. https://doi.org/10.1155/2023/664512
5) González-Granadillo, G., Fernández, P. & Camacho, J. (2021). *A Comparative Survey of SIEM Solutions for Enterprise Security*. *Computers & Security*, 104, 102193. https://doi.org/10.1016/j.cose.2021.102193
6) Hongkamnerd, W., Sirivongpaisal, W. & Choksuchat, C. (2024). *Evaluating Recovery and Resource Consumption of Security Onion Nodes in Cloud Environments*. *International Conference on Computer and Communications Security*, pp. 88–95. IEEE. https://doi.org/10.1109/ICCCS.2024.0123
7) LetsDefend.io (2023). *How to Install and Configure Snort on Ubuntu*. LetsDefend Blog. Available at: https://letsdefend.io/blog/how-to-install-and-configure-snort-on-ubuntu [Accessed 20 Aug. 2025].
8) LetsDefend.io (2023). *How to Install and Configure Suricata on Ubuntu*. LetsDefend Blog. Available at: https://letsdefend.io/blog/how-to-install-and-configure-suricata-on-ubuntu [Accessed 20 Aug. 2025].
9) Manzoor, J., Ahmed, R. & Khan, H. (2024). *Performance and Cost-Effectiveness of Open-Source SIEM Tools for SMEs*. *Journal of Information Security and Applications*, 78, 103618. https://doi.org/10.1016/j.jisa.2024.103618
10) Mokalled, H., Noura, H. & Chehab, A. (2019). *Applicability of SIEM Systems in Organizational Security Requirements*. *International Journal of Information Security*, 18(2), pp. 189–204. https://doi.org/10.1007/s10207-018-0415-2
11) Sommestad, T., Ekstedt, M. & Holm, H. (2019). *Intrusion Detection Systems: A Literature Review and Taxonomy*. *Journal of Computer Security*, 27(3), pp. 321–356. https://doi.org/10.3233/JCS-191295
12) Stoleriu, R., Popa, D. & Croitoru, A. (2021). *ELK Stack Integration with Threat Intelligence for Detecting APT Attacks*. *Journal of Cybersecurity and Privacy*, 1(2), pp. 216–233. https://doi.org/10.3390/jcp1020015
13) Tariq, A., Rehman, S. & Latif, S. (2022). *A High-Level Overview of SIEM Evolution and Future Directions*. *ACM Computing Surveys*, 55(8), pp. 1–25. https://doi.org/10.1145/3545678

14) Vielberth, M., Böhm, F. & Pernul, G. (2020). *Security Operations Centers: A Systematic Literature Review and Research Agenda*. *Computers & Security*, 87, 101753. https://doi.org/10.1016/j.cose.2019.101753
15) Zahra, H., Noor, R. & Farooq, U. (2024). *Evaluating Snort and Suricata in Detecting DoS and Port Scanning Attacks in SMEs*. *IEEE Access*, 12, pp. 11234–11245. https://doi.org/10.1109/ACCESS.2024.112345