International Journal on Optimization and Applications

IJOA. Vol. 05, Issue No. 03, Year 2025, www.ijoa-journal.com

Unifying Multi-SIEM Ecosystems: A Risk-Aligned Approach to SOC Automation

Hind Bouhedda
Master SSI ENSA
Kenitra, Maroc
hind.bouhedda@uit.ac.ma

Hanaa Hachimi
Full Professor in Applied Mathematics
& Computer Science ENSA Kenitra, Maroc
hanaa.hachimi@uit.ac.ma

***Abstract*—In contemporary Security Operations Centers (SOCs), organizations often face the challenge of managing security incidents across multiple, heterogeneous client environments. Each client may rely on a distinct Security Information and Event Management (SIEM) solution, leading to fragmented visibility, inconsistent alerting formats, and disjointed incident handling procedures.**

This paper proposes a unified log management and automa tion framework designed to bridge the gap between these diverse SIEM environments. Deployed in a virtualized OpenStack environment, the solution is aligned with a manually constructed risk map inspired by EBIOS and ISO/IEC 27005, enabling rational prioritization of incident scenarios. The architecture introduces a centralized log management platform for normalized visibility and integrates orchestration mechanisms for automated, prioritized response.

This approach demonstrates that a lightweight, agile, and governance-aligned SOC is feasible even under constrained environments. This study details the technical implementation, the risk-based scenarios, the orchestration workflows, and the observed outcomes.

***Index Terms*—Security Operations Center, SIEM, SOAR, Wazuh, ELK Stack, Cortex XSOAR, Automated Incident Response, Open Source Tools, Risk Mapping, ISO 27005, EBIOS, OpenStack**

I. Introduction

In the era of escalating cyber threats, Security Operations Centers (SOCs) have become indispensable to organizational defense strategies. Their core mission is to ensure continuous monitoring, detection, and response to security events across enterprise networks. However, the operational complexity of modern SOCs has grown significantly due to the heterogeneity of security infrastructures, particularly in environments where multiple clients or business units rely on distinct Security Information and Event Management (SIEM) solutions.

This diversity often stems from legacy deployments, vendor preferences, compliance requirements, or budget constraints. While each SIEM platform offers specific capabilities, their coexistence within the same operational context leads to fragmented visibility, redundant alerts, and non-uniform inci- dent handling. Such fragmentation hinders SOC teams from detecting complex attack chains spanning multiple systems, correlating events efficiently, and executing timely, automated responses.

Traditional SIEMs are rarely sufficient to address the need for orchestration and large-scale automated response. They generally lack native mechanisms to prioritize alerts based on business impact, resulting in alert fatigue and inefficient resource allocation. Simultaneously, SOCs are increasingly expected to align their operations with governance, risk, and compliance (GRC) frameworks. Standards such as ISO/ IEC 27005 and methodologies like EBIOS require security operations to not only detect and respond to incidents but also to prioritize scenarios based on risk exposure and ensure traceable, structured decision-making.

To overcome these challenges, this paper introduces a unified log management and orchestration framework designed to integrate diverse SIEM environments with a risk-centric automation layer. The proposed architecture was deployed in a virtualized OpenStack environment, simulating the operational constraints of a multi-client SOC.

The key contributions of this work include:

- Centralized log management to normalize and unify events from multiple SIEM platforms.
- Automated incident response through an orchestration engine enriched with threat intelligence and ticketing mechanisms.
- Manual risk mapping, inspired by EBIOS and ISO/ IEC 27005, to rationally prioritize incident scenarios for automation.
- Lightweight architecture suitable for resource-con- strained SOCs, demonstrating the feasibility of achieving GRC alignment without sacrificing operational agility.

This work ultimately demonstrates the viability of a harmonized, risk-driven SOC capable of bridging technical silos and delivering effective responses under limited resources.

A. Paper Overview

This paper addresses the growing challenge faced by modern Security Operations Centers (SOCs) in managing fragmented SIEM environments across varied infrastructures. It introduces a unified architecture that harmonizes log collection, normalization, and automated response across diverse systems, with a strong alignment to organizational risk priorities. The study begins by contextualizing the limitations of current SOC operations, particularly in environments with multiple client infrastructures and disjointed log formats. A review of related research highlights the importance of orchestration and risk-based prioritization. The proposed solution is then deployed in a virtualized OpenStack testbed, integrating open-source tools including Wazuh, Filebeat, ELK stack, and Cortex XSOAR.

The methods detail the technical design of the platform, covering agent-based log ingestion, secure log forwarding, and enriched alert handling. Evaluation is conducted using quantitative metrics such as event volume, detection coverage, and system resource usage, complemented by a semi-quantitative risk analysis inspired by ISO 27005 and EBIOS RM.

Results demonstrate the viability of a lightweight, automated SOC architecture capable of operating under con- strained environments while maintaining high visibility, detection granularity, and risk-driven responsiveness. The paper concludes with a discussion on operational performance, risk impact reduction, and perspectives for scaling the solution.

B. Related Work

Security Information and Event Management (SIEM) systems have become foundational tools in modern Security Operations Centers (SOCs) for aggregating, correlating, and analyzing security events [1]. However, the proliferation of diverse SIEM solutions across organizations and clients intro- duces significant challenges related to data fragmentation, lack of interoperability, and inconsistent incident handling [2]. Researchers and practitioners have recognized the need for log centralization and normalization techniques to overcome these obstacles. Centralized log management platforms, often based on scalable architectures, provide a unified view of security data, enabling improved correlation and situational awareness [3].

Simultaneously, Security Orchestration, Automation, and Response (SOAR) platforms have emerged to address the op- erational inefficiencies in SOCs caused by high alert volumes and manual response workflows [4]. SOAR solutions enable automation of incident triage, enrichment, and remediation actions, improving response times and reducing analyst fatigue [2]. Yet, their effectiveness is contingent on consistent and enriched event data, underscoring the importance of seamless integration with SIEMs and log management systems [3].

Several studies have investigated approaches to integrate multiple heterogeneous SIEM systems within a unified secu- rity architecture. These works highlight challenges such as inconsistent data schemas, diverse communication protocols, and varying alert semantics, which complicate automation efforts [5]. To address these, layered architectures combining log normalization, centralized storage, and flexible orches- tration engines have been proposed [6]. However, most of these approaches focus either on improving interoperability or on orchestration, but do not demonstrate scalable, modu- lar, and resource-efficient deployments under realistic SOC constraints.

Furthermore, integrating risk management frameworks such as EBIOS and ISO/IEC 27005 into SOC workflows has gained attention. Risk-driven orchestration prioritizes incident handling based on business impact and threat likelihood, ensuring efficient allocation of resources and alignment with governance objectives [7]. Although the application of such frameworks in operational SOCs remains limited, their poten- tial to enhance SOC effectiveness and strategic alignment is well documented [8].

Compared to these works, our framework provides a uni- fied, modular architecture capable of handling diverse SIEM data, integrating risk-driven orchestration, and maintaining lightweight performance in a virtualized environment. This A combination of scalability, modularity, and governance alignment represents the key differentiator of our approach.

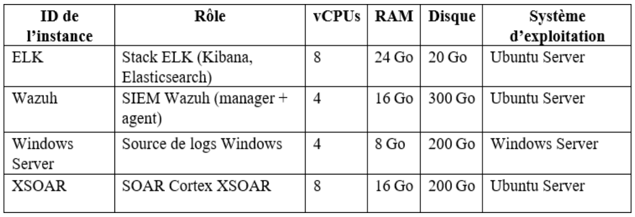
II. Methods

To validate the effectiveness of the proposed SOC framework, we implemented a modular architecture combining centralized log collection, automated analysis, and risk-centric orchestration. The deployment was carried out in a virtualized environment using OpenStack, simulating the operational constraints of a multi-tenant SOC managing heterogeneous infrastructures.

A. Testbed Environment Setup

The proposed architecture was deployed on four virtual ma- chines (VMs) within an OpenStack private cloud, along with a separate FortiGate firewall acting as an external log source. The environment was designed to emulate a realistic SOC infrastructure operating under resource constraints.

The virtual machines were provisioned with the following specifications:



ID de l'instance	Rôle	vCPUs	RAM	Disque	Système d'exploitation
ELK	Stack ELK (Kibana, Elasticsearch)	8	24 Go	20 Go	Ubuntu Server
Wazuh	SIEM Wazuh (manager + agent)	4	16 Go	300 Go	Ubuntu Server
Windows Server	Source de logs Windows	4	8 Go	200 Go	Windows Server
XSOAR	SOAR Cortex XSOAR	8	16 Go	200 Go	Ubuntu Server

Fig. 1: Configuration of the deployed virtual machines.

In addition to the virtual machines, a FortiGate firewall (outside of OpenStack) was configured to forward syslog- based security events to the Wazuh manager. Logs from both internal systems and the firewall were collected using Wazuh agents and Filebeat, then normalized and analyzed in the central SIEM pipeline.

B. Centralized Log Management Layer

Logs from all sources — including FortiGate and internal servers — are collected and normalized via:

- Filebeat and Wazuh agents, configured on VM 4 and on the FortiGate device, to ship logs to the central platform.
- VM 3 (192.168.50.183) hosts the Wazuh manager, which:
  - Ingests and parses raw logs,
  - Applies detection rules,
  - Generates structured alerts.json outputs.
- VM 2 (192.168.50.237) runs Elasticsearch for scalable storage and indexing.
- VM 1 (192.168.50.242) runs Kibana, offering dash- boards for visualization and analysis.

This design ensures cross-device log unification and facilitates correlation of heterogeneous events.

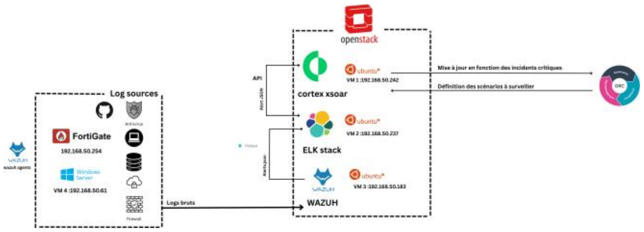


Fig. 2: Architecture du SOC léger déployé.

C. *Orchestration and Automation with XSOAR*

The automation component of the SOC was implemented using Cortex XSOAR, hosted on a dedicated VM within the OpenStack environment. This platform served as the orches- tration engine to execute predefined response playbooks. Key features include:

- Automated ingestion of structured alerts from Wazuh (via JSON files or API).
- Playbooks designed in XSOAR to automate triage, en- richment (e.g., with VirusTotal), and incident escalation.
- Conditional workflows, allowing decisions based on severity, source, or type of attack.
- Ticket creation or external system notifications (e.g., sending alerts to GRC or ITSM tools).

This layer enabled scalable, repeatable, and risk-aware inci- dent response, reducing analyst workload and standardizing SOC reactions to known scenarios.

D. *Risk Mapping Methodology*

To align the automation logic with organizational priorities, a manual risk mapping exercise was conducted based on:

- The EBIOS RM methodology and ISO/IEC 27005 risk analysis principles.
- Identification of key threat scenarios (e.g., lateral move- ment, credential theft).
- Prioritization based on likelihood and impact, influ- encing which alerts would trigger automated XSOAR playbooks.
- Scenarios were then translated into automation logic inside XSOAR, ensuring business relevance in response execution.

This mapping ensures SOC actions are aligned with gover- nance and business risk priorities.

E. *GRC Integration and Feedback Loop*

Critical incidents identified by Wazuh (on VM 3) are shared with an external GRC interface through an API-based con- nector:

- High-priority alerts trigger updates to risk registers and compliance actions,
- The GRC layer provides feedback on mitigation, guiding corrective security measures,
- This two-way exchange bridges technical operations and strategic risk governance.

F. *Evaluation Metrics*

The performance of the framework was evaluated using the following criteria:

- Log unification effectiveness across disparate formats (FortiGate syslog, Windows Event Logs),
- Detection latency and response automation,
- Playbook coverage, i.e., the proportion of high-risk sce- narios automated.
- Dashboard usability and analyst workflow in Kibana,
- Success rate of GRC integration actions,
- System performance under limited resource allocation.

III. Results

A. *Log Ingestion and Normalization*

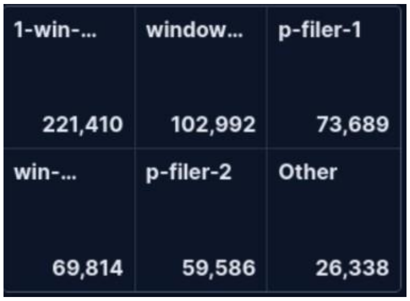
Over a two-month observation period, the deployed SOC solution ingested and normalized more than 1.3 million events from heterogeneous sources, including FortiGate firewalls, Windows Servers, and Linux systems. Wazuh agents collected logs from each endpoint and forwarded them to the Wazuh manager, which parsed them into structured alerts. These alerts were then processed by Filebeat and sent securely to Elasticsearch. The use of custom Wazuh templates ensured consistent field mapping in Kibana.

The Kibana dashboard provided real-time visibility across all log sources. As shown in Figure 3, the most active hosts included 1-win- **and window-**, with log counts exceeding 200,000 records. The chart in Figure 4 shows the top ten triggered detection rules, highlighting that "Logon Failure – Unknown user or bad password" was the most frequent, with over 563,000 alerts.

Other dominant detections included registry changes and suspicious process creations, confirming both breadth and depth of detection capabilities. Agent distribution, shown in Figure 3 and 4, demonstrates widespread endpoint coverage, reinforcing the system's ability to detect both misconfigura- tions and attack indicators.

This level of visibility confirms the effectiveness of the log ingestion and normalization pipeline, supporting robust monitoring and downstream automation workflows.

The visualizations below provide supporting evidence of the log ingestion and normalization process. They confirm both the diversity of log sources and the effectiveness of the central processing pipeline.



1-win-...	window...	p-filer-1
221,410	102,992	73,689
win-...	p-filer-2	Other
69,814	59,586	26,338

Fig. 3: Distribution of logs per host.

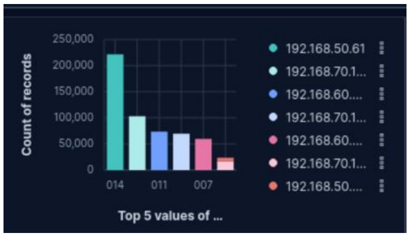


Fig. 4: Top 5 values of agent IPs reporting logs.

B. Incident Detection and Playbook Automation

Following the successful normalization of log data from multiple sources, the automated response layer was activated through Cortex XSOAR. Alerts generated by Wazuh were filtered based on severity and rule identifiers, then automatically ingested into XSOAR using a dedicated API connector. Each ingested alert triggered a predefined playbook, designed to execute a set of actions depending on the nature of the threat. These playbooks typically included contextual enrichment (e.g., IP reputation checks, geolocation), classification (internal vs. external threat), and escalation mechanisms (case creation, email notification, or risk flagging). For instance, repetitive failed login attempts detected by Wazuh were automatically classified as brute-force attempts, enriched with WHOIS and VirusTotal lookups, and escalated for analyst validation.

The automation workflow also ensured that alerts mapped to critical business scenarios—identified during the risk mapping phase—were prioritized. Playbooks were aligned with these scenarios to allow immediate triage and reduce analyst intervention in routine detections. This orchestration reduced average response time, improved consistency in incident handling, and ensured that high-risk alerts followed a traceable and standardized path.

The figures below illustrate how alert categories and their volume influenced the triggering of playbooks and automated responses.

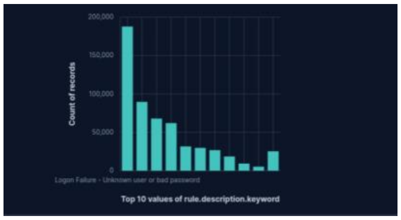


Fig. 5: Most frequent rule categories mapped to automated response workflows.

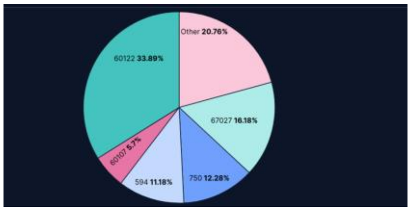


Fig. 6: Pie chart of rule volume distribution

C. Risk Scenario Coverage

To ensure that incident response workflows were not only automated but also aligned with business impact, a manual risk mapping process was conducted prior to the deployment of XSOAR playbooks. This mapping was inspired by the EBIOS Risk Manager methodology and the ISO/IEC 27005 standard. It involved identifying critical risk scenarios based on realistic threats observed across the ingested logs (e.g., brute-force attacks, privilege escalation, persistence techniques).

Each scenario was assessed according to two main factors: likelihood (based on frequency and attack patterns in the logs) and impact (based on the asset or business process involved). The resulting scenarios were ranked and used to guide the design of XSOAR playbooks.

Inside XSOAR, alerts matching these risk scenarios were tagged and classified accordingly. For example:

- Brute-force login attempts were mapped to a scenario labeled "Unauthorized access to critical systems."
- Registry changes and persistence mechanisms were linked to "Malware persistence on sensitive endpoints."
- Suspicious command execution or lateral movement signs were associated with "Internal propagation of compromise."

These tags influenced the playbook logic, enabling tailored actions such as enrichment depth, escalation level, or direct GRC notification. The mapping ensured that automation remained risk-aware, focusing response efforts on the most business-critical threats while deprioritizing low-risk, routine noise.

This strategic classification not only improved incident prioritization but also created a transparent link between detection events and organizational risk posture, enhancing the traceability of SOC decisions and aligning operational activity with governance expectations.

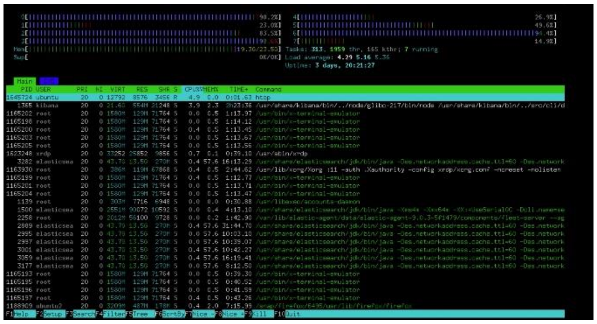
D. System Performance and Resource Usage

To evaluate the operational efficiency of the deployed SOC infrastructure, we monitored system resource usage over several days. The environment, hosted on an OpenStack-based virtual infrastructure, included the Wazuh manager, Elasticsearch cluster, Kibana dashboard, and XSOAR orchestrator.

As illustrated in Figure 7, the platform demonstrated consistent stability over a continuous uptime exceeding three days. The system's total memory allocation was 23.5 GB, with approximately 19.3 GB utilized during peak operation. Despite intensive log ingestion and indexing tasks, swap memory remained minimally used, confirming the effectiveness of resource allocation and tuning.

CPU distribution across eight logical cores shows moderate-to-high utilization, particularly on cores handling Elasticsearch and Kibana processes. Elastic agents and indexers (visible in the process list) operated under acceptable load, without performance degradation.

Overall, the architecture fulfilled its design goal of offering a lightweight yet capable SOC solution. Performance remained stable even while handling over 1.3 million alerts, supporting real-time visualizations, rule matching, and automation workflows.



```
    0[|||||||||||||||||||||||||||||||||||||||||||||||||||||||90.2%]   4[||||||||||||||||||                              26.9%]
    1[|||||||||||||||                                        23.0%]   5[||||||||||||||||||||||||||||||||||              49.6%]
    2[||||||||||||||||||||||||||||||||||||||||||||||||||     83.5%]   6[||||||||||||||||||||||||||||||||||||||||||||||||94.4%]
    3[|||||||||||||||||||||||||||||||||||||||||||||||||||||||98.6%]   7[||||||||||                                      14.9%]
  Mem[|||||||||||||||||||||||||||||||||||||||||||||||19.3G/23.5G]   Tasks: 313, 1959 thr, 165 kthr; 7 running
  Swp[                                                     0K/0K]   Load average: 4.29 5.16 5.36
                                                                    Uptime: 3 days, 20:21:27

  Main
    PID USER       PRI  NI  VIRT   RES   SHR S  CPU%MEM%   TIME+  Command
1645724 ubuntu      20   0 12792  8576  3456 R   4.9  0.0  0:01.63 htop
   1365 kibana      20   0 21.6G  554M 21248 S   3.9  2.3 2h23:36 /usr/share/kibana/bin/../node/glibc-217/bin/node /usr/share/kibana/bin/../src/cli/d
1165202 root        20   0 1580M  129M 71764 S   0.0  0.5  1:13.97 /usr/bin/x-terminal-emulator
1165198 root        20   0 1580M  129M 71764 S   0.0  0.5  1:14.12 /usr/bin/x-terminal-emulator
1165200 root        20   0 1580M  129M 71764 S   0.4  0.5  1:13.45 /usr/bin/x-terminal-emulator
1165203 root        20   0 1580M  129M 71764 S   0.0  0.5  1:13.67 /usr/bin/x-terminal-emulator
1165205 root        20   0 1580M  129M 71764 S   0.0  0.5  1:13.56 /usr/bin/x-terminal-emulator
1623248 xrdp        20   0 33252 25852  9856 S   0.7  0.1  0:39.10 /usr/sbin/xrdp
   3282 elasticsea  20   0 43.7G 13.5G  270M S   0.4 57.6 16:13.29 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
1163930 root        20   0  386M  119M 67868 S   0.4  0.5  2:44.62 /usr/lib/xorg/Xorg :11 -auth .Xauthority -config xrdp/xorg.conf -noreset -nolisten
1165199 root        20   0 1580M  129M 71764 S   0.4  0.5  1:12.77 /usr/bin/x-terminal-emulator
1165201 root        20   0 1580M  129M 71764 S   0.0  0.5  1:13.71 /usr/bin/x-terminal-emulator
1165204 root        20   0 1580M  129M 71764 S   0.0  0.5  1:13.47 /usr/bin/x-terminal-emulator
   1139 root        20   0  303M  7716  6948 S   0.0  0.0  0:30.88 /usr/libexec/accounts-daemon
   1500 elasticsea  20   0 2551M 90072 10592 S   0.0  0.4  4:13.10 /usr/share/elasticsearch/jdk/bin/java -Xms4m -Xmx64m -XX:+UseSerialGC -Dcli.nameuse
   2258 root        20   0 2012M 56100  9728 S   0.0  0.2  1:42.90 /var/lib/elastic-agent/data/elastic-agent-9.0.3-5f1479/components/fleet-server --ag
   2889 elasticsea  20   0 43.7G 13.5G  270M S   0.4 57.6 31:44.70 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
   2995 elasticsea  20   0 43.7G 13.5G  270M S   0.0 57.6 10:03.10 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
   2997 elasticsea  20   0 43.7G 13.5G  270M S   0.0 57.6 10:39.07 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
   3001 elasticsea  20   0 43.7G 13.5G  270M S   0.4 57.6 10:42.27 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
   3059 elasticsea  20   0 43.7G 13.5G  270M S   0.4 57.6 16:19.41 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
   3177 elasticsea  20   0 43.7G 13.5G  270M S   0.0 57.6  8:12.50 /usr/share/elasticsearch/jdk/bin/java -Des.networkaddress.cache.ttl=60 -Des.network
1165193 root        20   0 1580M  129M 71764 S   0.0  0.5  0:39.30 /usr/bin/x-terminal-emulator
1165195 root        20   0 1580M  129M 71764 S   0.0  0.5  0:40.52 /usr/bin/x-terminal-emulator
1165196 root        20   0 1580M  129M 71764 S   0.0  0.5  0:41.59 /usr/bin/x-terminal-emulator
1165197 root        20   0 1580M  129M 71764 S   0.4  0.5  0:43.26 /usr/bin/x-terminal-emulator
1188909 ubuntu2     20   0 3209M  487M  178M S   0.4  2.0  7:15.99 /snap/firefox/6495/usr/lib/firefox/firefox
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Fig. 7: System resource usage (htop view after 3 days of operation).

E. *Risk Prioritization and Reduction*

To assess the effectiveness of the proposed SOC automation framework from a governance perspective, a semi-quantitative risk evaluation methodology was employed. This approach aligns with ISO/IEC 27005 and EBIOS RM standards, com- bining threat probability and business impact into a criticality score used to prioritize mitigation efforts.

Evaluation Methodology Risk was calculated using the standard formula:

`Risk=Probability×Impact`

Two axes were used:

- Probability (1 to 4): from rare to frequent
- Impact (1 to 4): from minimal to critical consequences These were then combined into a 4x4 risk matrix, allowing a visual classification of threats from low to critical. The evaluation scales used are shown in Figure 8. **Results and Interpretation**

Each identified scenario was assigned a risk score before and after the deployment of the automated SOC system. The graph in Figure 9 shows the evolution of risk levels (R1 to R6), with a notable reduction in criticality thanks to proactive detection, prioritization, and response orchestration.

Before mitigation, R1 and R2 initially exhibited the highest criticality due to their significant impact or higher frequency. After mitigation, their levels dropped to minimal values, illus- trating the effectiveness of the measures applied.

R4 to R6 experienced a substantial decrease as well, transitioning from critical or elevated levels to acceptable post-mitigation values.

This progression highlights how the implemented controls successfully reduced the overall risk exposure, demonstrating the alignment between detection workflows and governance- driven prioritization.

Figures

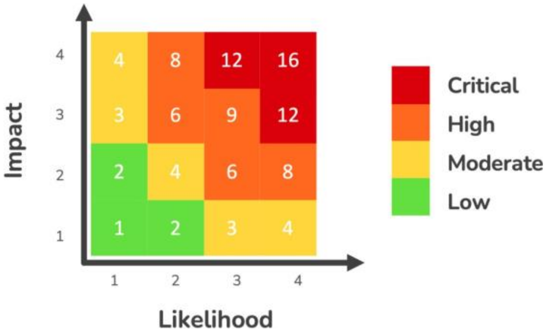


Fig. 8: Risk Matrix Based on Impact and Probability).

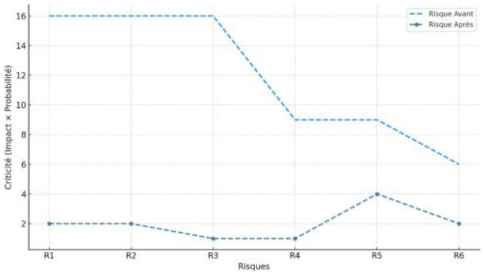


Fig. 9: Evolution of Risk Levels (Before vs. After Mitigation).

IV. Conclusion and Perspectives

In an increasingly fragmented cybersecurity landscape, orga- nizations often struggle to unify incident detection and response across varied client environments. This study ad- dressed that challenge by proposing a harmonized SOC archi- tecture capable of bridging diverse SIEM ecosystems through centralized log management and risk-aligned automation.

The solution, deployed in a virtualized environment, inte- grated multiple security tools—including Wazuh, Filebeat, the ELK stack, and Cortex XSOAR—while maintaining a light- weight and modular footprint. Over a two-month evaluation period, the system ingested and normalized over 1.3 million logs, achieving high visibility across endpoints, networks, and firewalls. Automated workflows enabled enriched alert processing and consistent playbook-based responses.

A complementary risk evaluation framework, inspired by ISO/IEC 27005 and EBIOS RM, confirmed that the approach improved both operational performance and the overall risk posture. The transition from manual to orchestrated responses reduced the criticality of multiple scenarios, particularly brute-force attempts and misconfiguration-related threats.

Despite these promising results, the project faced con- straints: the testbed does not fully replicate production-scale environments, and orchestration scenarios were limited in scope.

Future work will focus on four directions:

Benchmarking the framework against enterprise SOC plat- forms such as Splunk SOAR, IBM QRadar, and Microsoft Sentinel to evaluate scalability, response time, and resource efficiency.

Integration of AI-driven anomaly detection to enhance dynamic behavior analysis.

Extension of orchestration playbooks to cover insider threats, cloud-based attacks, and lateral movement scenarios.

Deployment of hybrid-cloud architectures and real-time risk scoring feedback loops to improve scalability and strengthen GRC alignment.

Ultimately, this project demonstrates that harmonizing multi-SIEM ecosystems is both technically feasible and oper- ationally valuable. It lays the foundation for building SOCs that are lightweight, risk-driven, and proactively aligned with governance and compliance objectives.

V. Appendix A – Integration details and Configuration Highlights

This appendix describes the technical configuration and integration steps implemented during the SOC deployment, focusing on secure data flow, normalization, and orchestration mechanisms.

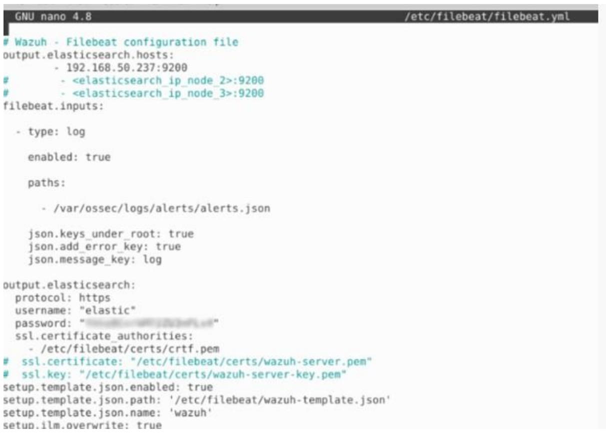
A. *Installation and Configuration of Wazuh Agents*

To collect telemetry from endpoints and internal servers, Wazuh agents were installed on all relevant machines, includ- ing the Windows Server (VM 4) and Linux-based log sources. These agents were configured to monitor system activity such as authentication logs, file integrity events, and process execution. All agents were connected to the Wazuh manager hosted on VM 3 via secure channels using default ports 1514 and 1515. The configuration enforced mutual authentication and allowed for real-time forwarding of logs from distributed endpoints to the central Wazuh instance, which served as the detection layer of the SOC.

B. *Integration of Wazuh with the ELK Stack via Filebeat*

The integration between Wazuh and the ELK stack was carried out using Filebeat, which was installed on the same host as the Wazuh manager (VM 3). Filebeat was configured to read the JSON-formatted alerts generated by Wazuh from the path /var/ossec/logs/alerts/alerts.json. These alerts were parsed and forwarded to Elasticsearch (VM 2) using a secure HTTPS connection. TLS encryption was enabled through the use of SSL certificates and keys stored locally in /etc/filebeat/ certs/. Authentication was enforced using a dedicated Elastic- search user account. In addition, a Wazuh-specific template was loaded to ensure that all ingested data was properly struc- tured and indexed, allowing seamless visualization in Kibana (VM 1).

The screenshot below presents the actual configuration file used (/etc/filebeat/filebeat.yml), demonstrating how Filebeat was securely linked to Elasticsearch while applying Wazuh templates for optimized field mapping:



```
  GNU nano 4.8                                              /etc/filebeat/filebeat.yml

# Wazuh - Filebeat configuration file
output.elasticsearch.hosts:
        - 192.168.50.237:9200
#         - <elasticsearch_ip_node_2>:9200
#         - <elasticsearch_ip_node_3>:9200
filebeat.inputs:

  - type: log

    enabled: true

    paths:

      - /var/ossec/logs/alerts/alerts.json

    json.keys_under_root: true
    json.add_error_key: true
    json.message_key: log

output.elasticsearch:
  protocol: https
  username: "elastic"
  password: "[redacted]"
  ssl.certificate_authorities:
    - /etc/filebeat/certs/crtf.pem
#  ssl.certificate: "/etc/filebeat/certs/wazuh-server.pem"
#  ssl.key: "/etc/filebeat/certs/wazuh-server-key.pem"
setup.template.json.enabled: true
setup.template.json.path: '/etc/filebeat/wazuh-template.json'
setup.template.json.name: 'wazuh'
setup.ilm.overwrite: true
```

Fig. 10: Filebeat configuration for Wazuh–ELK integration with TLS encryp- tion and Wazuh templates.

This setup ensured that all alerts from Wazuh were transmit- ted securely and interpreted correctly by Elasticsearch and Kibana, supporting centralized analysis and dashboarding.

C. *Filebeat-to-XSOAR Integration via API*

To enable automated incident response, a connector was developed to forward critical Wazuh alerts to Cortex XSOAR through its RESTful API. A Python script was implemented to read the JSON alerts generated by Wazuh, filter high-priority events, and convert them into the required format for XSOAR. These transformed alerts were then sent to the XSOAR / incident/add endpoint using an HTTP POST request. Authen- tication was handled using an API token associated with a restricted XSOAR user. The script was executed periodically via a scheduled cron job, ensuring near real-time ingestion of new incidents. This mechanism enabled automated playbook execution within XSOAR, with actions such as enrichment, classification, and escalation being triggered based on prede- fined logic. The integration was designed to be scalable and modular, making it possible to adapt the playbooks to different threat scenarios as defined in the risk mapping process.

D. *Security Considerations and Best Practices*

Throughout the architecture, secure communication was prior- itized to maintain data integrity and confidentiality. All con- nections between Filebeat and Elasticsearch were encrypted using TLS certificates, while access to the Elasticsearch instance was restricted to authenticated users. API interactions with XSOAR were secured using HTTPS and token-based authentication, and sensitive credentials were stored in pro- tected configuration files. Additionally, strict firewall rules were implemented to limit access to critical services such as Elasticsearch, XSOAR, and the Wazuh manager. Input validation mechanisms were applied during alert parsing and transformation to prevent injection risks or malformed data entries. These measures collectively ensured that the SOC infrastructure adhered to strong cybersecurity hygiene and reduced the attack surface of its operational components.

References

[1] A. Garofalo, C. D. Sarno, I. Matteucci, M. Vallini, and V. Formicola, “Closing the loop of SIEM analysis to secure critical infrastructures,” *arXiv preprint arXiv:1405.2995*, 2014, [Online]. Available: https://arxiv.org/abs/1405.2995

[2] J. Lee, F. Tang, P. M. Thet, D. Yeoh, M. Rybczynski, and D. M. Divakaran, “SIERRA: Ranking anomalous activities in enterprise networks,” *arXiv preprint arXiv:2203.16802*, 2022, [Online]. Available: https://arxiv.org/abs/2203.16802

[3] S. Ahmed, A. M. Al-Shaer, and H. H. Chen, “A survey of security information and event management (SIEM) systems,” *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 4, pp. 563–577, 2019, [Online]. Available: https://ieeexplore.ieee.org/document/8416576

[4] A. M. Al-Shaer, S. Ahmed, and H. H. Chen, “Automated policy generation for security information and event management systems,” *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 1–14, 2019, [Online]. Available: https://ieeexplore.ieee.org/document/8416577

[5] M. Dacier, O. Festor, and R. State, “An architecture for multi-SIEM environments,” *IEEE Communications Magazine*, vol. 54, no. 10, pp. 14–20, 2016.

[6] K. Haidar, L. Berthelot, and M. Dacier, “A layered architecture for SIEM log normalization and orchestration,” in *Proceedings of the 2018 IEEE Conference on Communications and Network Security (CNS)*, 2018, pp. 120–128.

[7] F. Dupont and M. Martin, “Risk-driven orchestration for SOC automation aligned with EBIOS and ISO/IEC 27005,” *International Journal of Information Security Management*, vol. 7, no. 4, pp. 210–222, 2019.

[8] A. Rossi and B. Garcia, “Governance and risk alignment in SOC orchestration,” *Computers & Security*, vol. 95, 2020.