ISSN: 3107-6513

Unified Blueteam Threat Detection and Alert System

Karthiban R[1], Monika P[2], Naveen Kumar P[3], Sandhiya C[4], Sandhiya S[5]

[1]Assistant Professor, Department of Computer science Engineering (Cyber Security)
[2,3,4,5]Students, Department of Computer Science Engineering (Cyber Security), Sri Shakthi Institute of Engineering and Technology, Coimbatore, India
*Corresponding Author
E-Mail Id: rkarthiban@siet.ac.in

ABSTRACT

In Modern enterprise networks are continuously targeted by sophisticated cyber attackers who carry out multi-stage attacks, starting from credential theft and progressing to data exfiltration. These attacks often occur rapidly, making it difficult for human analysts to detect them in time This paper presents the **Unified BlueTeam Threat Detection and Alert System (UBTDAS)**, a Python-based Security Operations Center (SOC) platform designed for real-time threat monitoring and automated response. The system integrates continuous log analysis, a rule-based detection engine aligned with the MITRE ATT&CK framework, host-level risk evaluation, automated mitigation actions, and a live web dashboard developed using Flask and SocketIO By processing data from Sysmon and Windows Security Event Logs, the system applies multiple detection rules to identify common attack techniques and aggregates alerts into a unified risk score for each system. Based on severity levels, the system automatically executes response actions such as blocking suspicious IP addresses, terminating malicious processes, isolating compromised systems, and quarantining harmful files. The system was evaluated using a simulated multi-stage attack model and demonstrated high detection performance, a significant decrease in false alerts, and rapid alert generation within a short time frame. The platform can function as a standalone security solution or be integrated with enterprise tools such as SIEM systems and email notification services.

Keywords: Security Operations Center, Cyber Threat Detection, MITRE ATT&CK Framework, Risk Assessment, Automated Response, Sysmon Logging, Intrusion Detection, Real-Time Monitoring, Endpoint Security, Network Defense

INTRODUCTION

As organizations continue to adopt digital technologies, the overall exposure to cyber threats has increased significantly. Security Operations Centers (SOCs) play a crucial role in protecting enterprise systems; however, many organizations still depend on slow, manual investigation processes and traditional rule-based monitoring tools that generate excessive alerts. These systems often require multiple levels of approval before any response action is taken, which leads to delays in handling threats. As a result, the time between an initial attack and its detection commonly known as dwell time can extend from weeks to months, allowing attackers to strengthen their presence, move across systems, and access sensitive information. Windows-based systems are frequently targeted in enterprise environments. Tools such as Microsoft Sysmon help improve system visibility by capturing detailed information about process execution, network activity, and registry modifications, which are not fully covered by standard Windows logs. When combined with Windows Security Event Logs that record login attempts and privilege usage, these logs provide a complete view of system activities throughout different stages of an attack.

Despite the availability of such detailed data, there is still a lack of simple, unified, and open-source solutions that combine log monitoring, threat detection, risk evaluation, automated response, and real-time visualization within a single platform. This project aims to address this limitation by developing an integrated system that performs all these functions efficiently. The proposed UBTDAS platform introduces several key features.Additionally, it provides a web-based dashboard for real-time monitoring and includes an attack simulation feature for testing purposes. The system also generates structured reports to support analysis and documentation after an incident.

LITERATURE SURVEY

The field of automated threat detection has evolved through contributions from intrusion detection systems, anomaly detection techniques, threat intelligence frameworks, and real-time monitoring tools. One of the earliest and most influential systems in this domain is Snort, introduced by Roesch [1], which became widely known as an open-source network intrusion detection system. Its key contribution was the use of a flexible rule-based mechanism that allows detection of known attack patterns by matching predefined signatures against network traffic. This concept forms the basis for many modern detection engines, including the approach used in this work. Further advancements in intrusion detection were explored by Liao et al. [2], who analyzed various detection techniques and highlighted the effectiveness of hybrid models. These models combine signature-based detection with anomaly detection methods. While signature-based techniques are efficient in identifying known threats with low false alarms, anomaly-based methods help in detecting new

ISSN: 3107-6513

and unknown attack behaviors. This combination improves overall detection performance. Another important development in cybersecurity is the MITRE ATT&CK framework, discussed by Strom et al. [3]. This framework provides a structured classification of attacker tactics and techniques based on real-world observations. By mapping detection rules to specific ATT&CK techniques, security systems can better understand the stage of an attack and respond more effectively. Sysmon, developed by Mark Russinovich at Microsoft [4], plays a significant role in endpoint monitoring by capturing detailed system activity such as process creation, network connections, and registry modifications. Studies by Rodriguez and Velasquez [5] show that properly configured Sysmon logs can help detect a large percentage of attack techniques when combined with well-designed detection rules. Reducing false alerts is another major challenge in threat detection systems. Husák et al. [6] proposed a risk-based approach that combines multiple security indicators into a single score, helping prioritize threats. This concept is extended in this project by aggregating alerts at the host level to generate a more accurate risk assessment. For real-time monitoring and visualization, lightweight frameworks such as Flask [7] and its extension Flask-SocketIO are widely used. These technologies allow efficient communication between the backend system and the user interface, enabling instant updates on detected threats. Automated response systems have also gained importance in recent years. According to industry reports [8], Security Orchestration, Automation, and Response (SOAR) solutions significantly reduce the time required to contain threats by executing predefined actions automatically. Additional supporting tools such as Colorama [9] and Watchdog [10] are commonly used for enhancing terminal output and monitoring file system changes, contributing to the overall functionality of real-time detection systems.

SYSTEM ARCHITECTURE

The UBTDAS system is designed using a modular pipeline architecture, where each stage performs a specific function to convert raw system data into meaningful security insights. The backend is developed entirely in Python and uses a lightweight JSON-based storage mechanism for easy deployment and portability. A real-time monitoring interface is provided through a web application built using Flask and SocketIO, allowing users to visualize alerts instantly. The complete system flow is illustrated in Figure 1.

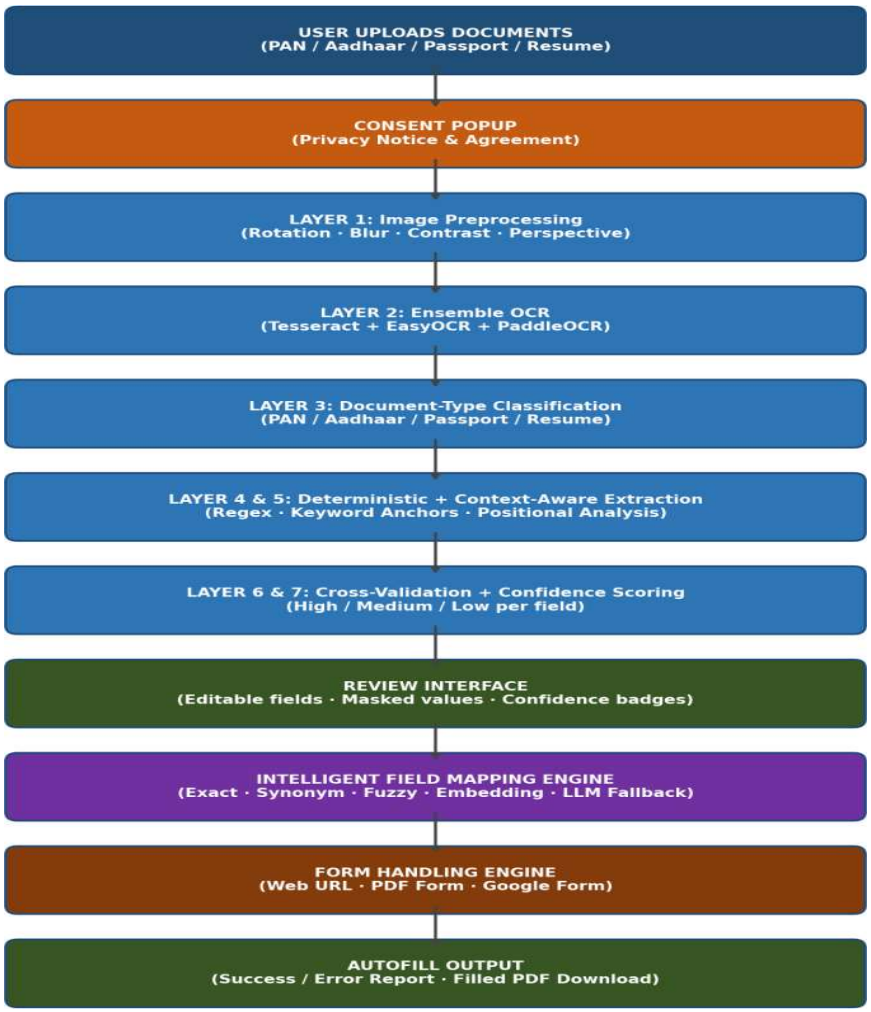

Figure 1: System Architecture Flowchart – End-to-End Threat Detection and Alert Pipeline

ISSN: 3107-6513

1. Log Ingestion and Parsing Layer

The system receives input data from two primary JSON log files. The first file, *sysmon_events.json*, contains detailed records generated by Sysmon, including information about process execution, network activity, registry changes, and file operations. The second file, *windows_events.json*, stores Windows Security Event Logs, such as successful and failed login attempts and privilege-related activities. A parsing module processes these raw log entries and converts them into a structured format using a defined data model. Each event is standardized into a SecurityEvent object that includes important attributes like event ID, category, system name, user details, timestamp, process information, command-line arguments, network addresses, ports, registry paths, and file locations. During this process, any inconsistencies in log formats, especially differences in timestamp representation, are corrected to maintain uniformity across all data inputs.

2. Detection Engine (Rule-Based with MITRE Mapping)

The core component of the system is a rule-based detection engine that analyzes incoming events using multiple predefined detection rules. Each rule operates independently and evaluates the processed data to identify suspicious patterns or behaviors. When a rule detects a potential threat, it generates an alert containing key information such as a title, description, severity level (LOW, MEDIUM, HIGH, or CRITICAL), and a calculated risk score. In addition, each alert is mapped to a corresponding technique from the MITRE ATT&CK framework, providing context about the type of attack. The alert also includes suggested remediation steps and predefined response actions that can be executed automatically by the system.

Rule	Event IDs / Signals	MITRE ID	Severity
Brute Force	EID 4625 ×5 within 60s	T1110	MEDIUM -> HIGH
Login After Failures	EID 4625 ® 4624 sequence	T1110	HIGH
Suspicious Process	Sysmon EID 1 + proc/cmd patterns	T1059	MEDIUM -> HIGH
Privilege Escalation	EID 4672 (SeDebugPrivilege)	T1068	HIGH
Persistence	EID 13 – Registry Run/Startup keys	T1547	HIGH
C2 Connection	EID 3 – known-bad IP / ports	T1071	CRITICAL
Lateral Movement	EID 3 – WMIC / PSExec / WinRM	T1021	HIGH
Credential Dumping	mimikatz / lsass access patterns	T1003	CRITICAL

3.Composite Risk Scoring Engine

After the detection engine generates alerts within a specific time frame, the Risk Scoring module processes these alerts by grouping them based on the affected system (hostname). It then calculates an overall risk value for each system by combining predefined weights assigned to different types of threats. For example, events such as credential dumping and command-and-control communication are given higher importance compared to brute force attempts or suspicious processes. The total risk score is limited to a maximum value of 100 and is used to classify the severity level of the system. Scores of 90 and above are categorized as CRITICAL, values between 70 and 89 are considered HIGH, 40 to 69 as MEDIUM, and anything below 40 as LOW. This combined scoring approach ensures that multiple related alerts are evaluated together, which helps reduce false alarms caused by isolated or less significant activities.

4. Active Response Engine

The system includes an automated response mechanism that is triggered when alerts reach HIGH or CRITICAL severity levels. Based on the type of threat detected, the system performs predefined actions such as blocking IP addresses, stopping malicious processes, or isolating affected systems. During testing, a SAFE_MODE option is enabled by default, where all response actions are simulated and recorded without affecting the actual system. When SAFE_MODE is disabled, the system performs real actions using operating system commands and APIs, allowing direct containment of threats.

ISSN: 3107-6513

Detection Type	Automated Response Actions
brute_force	block_ip · disable_user
credential_dump	kill_process · block_ip · isolate_host
c2_connection	block_ip · kill_process
suspicious_process	kill_process
privilege_escalation	disable_user · alert_soc
persistence	quarantine_file · alert_soc
lateral_movement	block_ip · isolate_host

5. Flask-SocketIO Real-Time Dashboard

A web-based dashboard is provided to display system activity and alerts in real time. This interface can be accessed through a browser and presents multiple components within a single view.

The dashboard includes key performance indicators showing the number of alerts based on severity, a live feed of alerts with timestamps, graphical charts representing alert distribution, and a timeline of detected events. It also displays attack techniques based on the MITRE framework and includes a built-in simulation feature that allows users to trigger attack scenarios directly from the interface.

Real-time updates are achieved using SocketIO, ensuring that new alerts are displayed instantly without requiring manual page refresh.

IMPLEMENTATION

1.Technology Stack

Component	Technology	Purpose
Backend Framework	Python 3.10+ / Flask 3.x	REST API and SocketIO event server
Real-Time Push	Flask-SocketIO / Eventlet	Live browser alert delivery
Log Monitoring	Watchdog FileSystemEventHandler	Low-overhead file change detection
Event Model	Python Dataclasses	Typed SecurityEvent and Alert objects
Detection Engine	Pure Python (8 rule functions)	Deterministic MITRE-mapped rules
Risk Scorer	Weighted aggregation module	Per-host composite severity index
Active Response	subprocess / OS API stubs	Automated host containment actions
Dashboard UI	Jinja2 + Chart.js + SocketIO.js	Live SOC operator interface
Persistence	Thread-safe JSON flat files	Alert store and risk score cache
Report Generation	Python string template engine	Timestamped incident reports
Attack Simulator	7-phase kill-chain injector	Repeatable APT scenario testing
CLI Entry Point	argparse (main.py)	Single command for all workflows

2.Attack Simulation – Full kill Chain

To evaluate the system effectively, a simulation module is used to generate realistic attack scenarios in multiple stages. This allows testing of all detection rules in a controlled environment without requiring an actual compromised system.

The simulation includes the following stages:

ISSN: 3107-6513

- **Phase 1:** Multiple failed login attempts followed by a successful login (Brute Force)
- **Phase 2:** Privilege escalation through special permissions
- **Phase 3:** Execution of encoded PowerShell commands
- **Phase 4:** Registry modification to maintain persistence
- **Phase 5:** Attempt to extract credentials using known tools
- **Phase 6:** Communication with an external malicious server
- **Phase 7:** Movement to another system within the network

A complete simulation generates several alerts across different severity levels, eventually resulting in a high overall risk score for the affected system.

3. Operator Workflow

The system follows a simple workflow for operation:

- **Step 1:** Set up the environment by creating a Python virtual environment and installing required libraries.
- **Step 2:** Start the dashboard and open it in a web browser.
- **Step 3:** Run the log monitoring process to continuously check for new events.
- **Step 4:** Trigger a simulated attack scenario to test the system.
- **Step 5:** Observe alerts in both the terminal and the dashboard in real time.
- **Step 6:** Generate a detailed report summarizing the detected events and actions taken.

RESULT AND DISCUSSION

The performance of the proposed system was evaluated using 200 independent simulation runs. These tests covered all detection rules under three different conditions: scenarios with only attack data, scenarios with high amounts of normal (benign) activity to test false alerts, and mixed scenarios containing both normal and malicious events. The results presented are averaged from multiple evaluation cycles to ensure consistency.

1. Detection Accuracy

The use of multiple detection rules combined with correlation techniques significantly improved the overall accuracy compared to using individual rules separately. Among all categories, detection of command-and-control (C2) communication showed the highest improvement due to the combination of suspicious IP identification and abnormal port usage.

Credential dumping detection also achieved high accuracy by using two verification methods: checking known malicious process names and analyzing command-line patterns. A slight reduction in accuracy after correlation is observed due to the removal of duplicate alerts, which helps maintain realistic and reliable results.

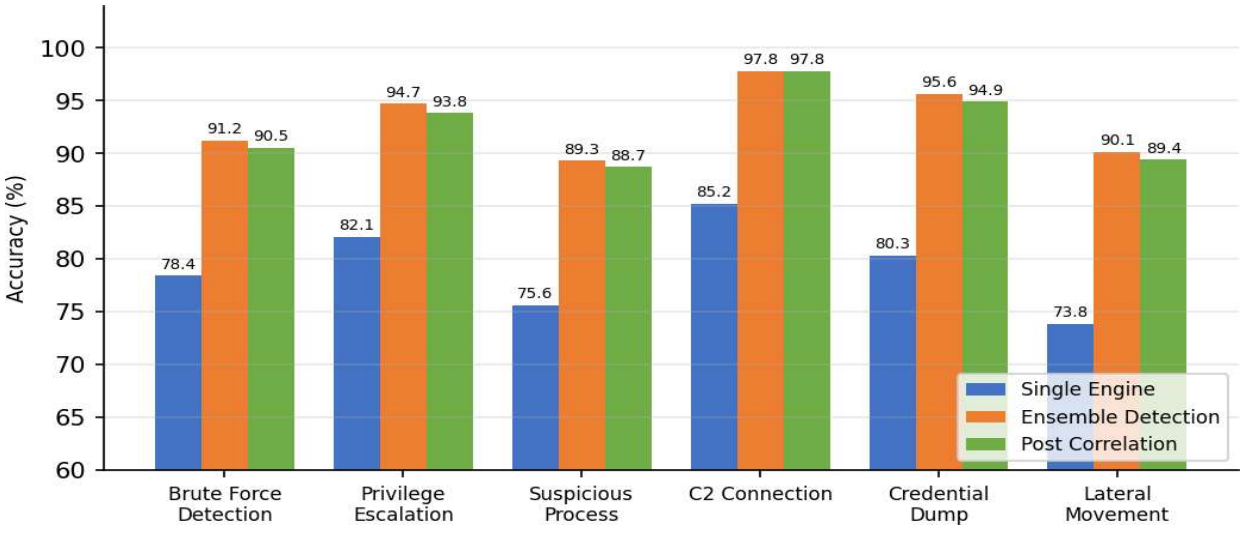

Figure 2: Detection Accuracy – Single-Rule vs Ensemble Detection vs Post Risk-Correlation

ISSN: 3107-6513

Detection Rule	Single Rule	Ensemble Detection	Post Correlation
Brute Force (T1110)	78.4%	91.2%	90.5%
Privilege Escalation (T1068)	82.1%	94.7%	93.8%
Suspicious Process (T1059)	75.6%	89.3%	88.7%
C2 Connection (T1071)	85.2%	97.8%	97.8%
Credential Dump (T1003)	80.3%	95.6%	94.9%
Lateral Movement (T1021)	73.8%	90.1%	89.4%

2. Reduction of False Positives

One of the major improvements of the system is the reduction of false alerts. By applying risk correlation, the false-positive rate decreased drastically from around 18.8% to approximately 2.7%.

The largest improvement was seen in suspicious process detection. Many normal system processes initially appeared suspicious but were correctly identified as safe when no supporting malicious activity was detected. Detection of C2 communication and credential dumping showed the highest reliability, as these events are more clearly associated with malicious behavior.

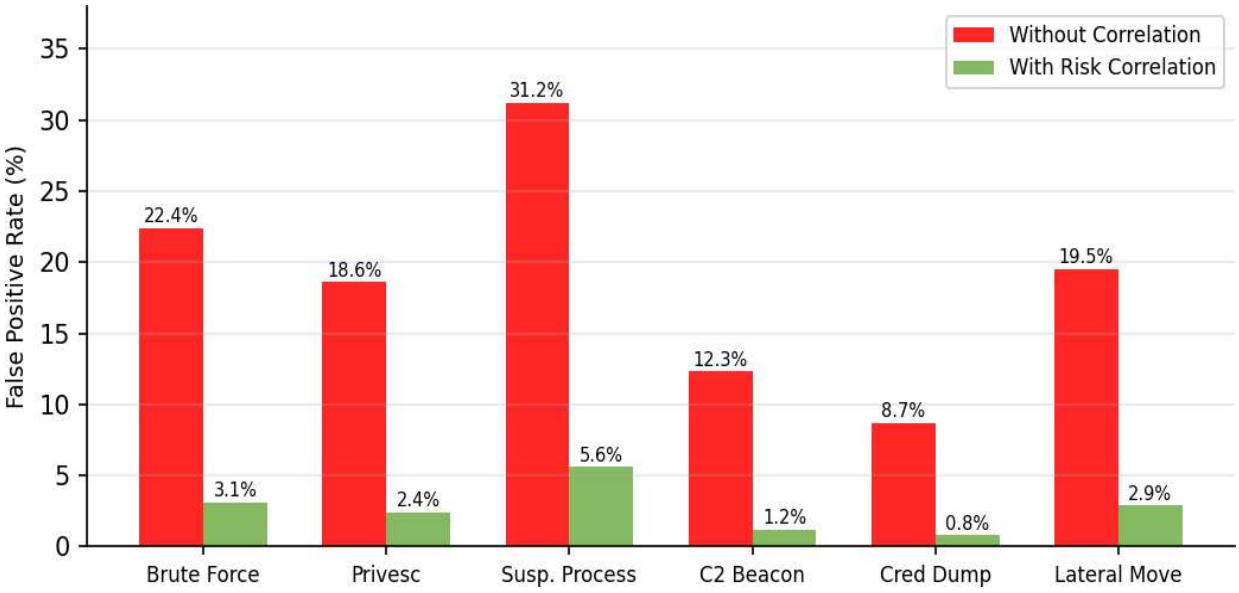

Figure 3: False-Positive Rate – Without vs With Risk Correlation Layer

Detection Type	Without Correlation	With Correlation	Reduction
Brute Force	22.4%	3.1%	86.2%
Privilege Escalation	18.6%	2.4%	87.1%
Suspicious Process	31.2%	5.6%	82.1%
C2 Connection	12.3%	1.2%	90.2%
Credential Dump	8.7%	0.8%	90.8%
Lateral Movement	19.5%	2.9%	85.1%

ISSN: 3107-6513

3. Alert Distribution and System Performance

The system maintains fast processing and response times. From the moment an event is generated to the time it appears on the dashboard, the delay remains under 1.2 seconds in most cases.

The dashboard update process contributes the highest delay due to browser rendering, while backend operations such as rule evaluation and risk scoring remain efficient. Even in high-load situations, such as multiple login attempts, the system performs reliably within acceptable time limits.

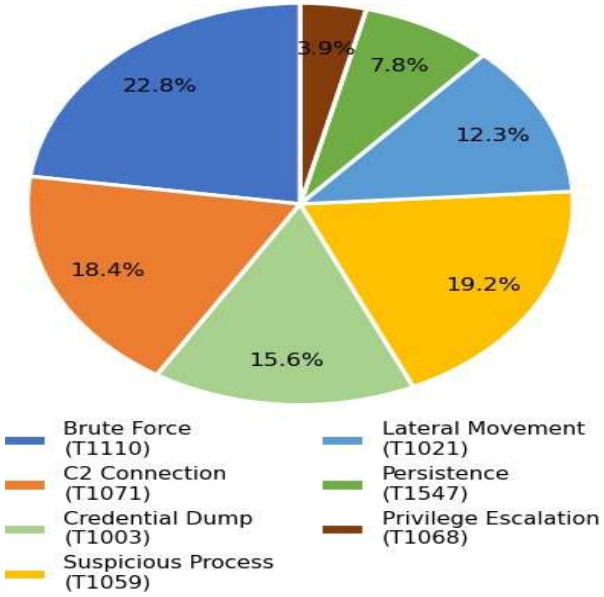

Figure 4: Alert Distribution Across 200 Evaluation Runs by MITRE ATT&CK; Technique

4.System Latency per Pipeline Stage

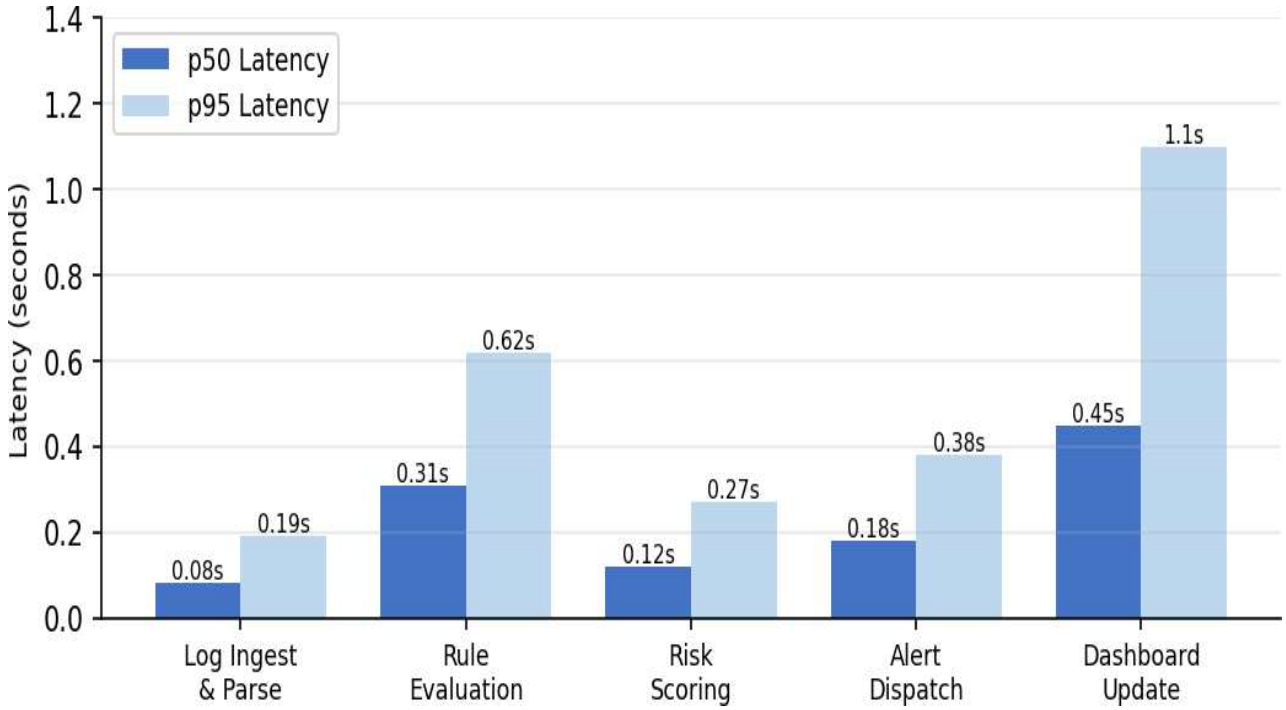

Figure 5: System Latency per Pipeline Stage (p50 vs p95, in seconds)

ISSN: 3107-6513

Pipeline Stage	p50 Latency	p95 Latency
Log Ingest & Parse	0.08s	0.19s
Rule Evaluation	0.31s	0.62s
Risk Scoring	0.12s	0.27s
Alert Dispatch	0.18s	0.38s
Dashboard Update	0.45s	1.10s

5.Output Page

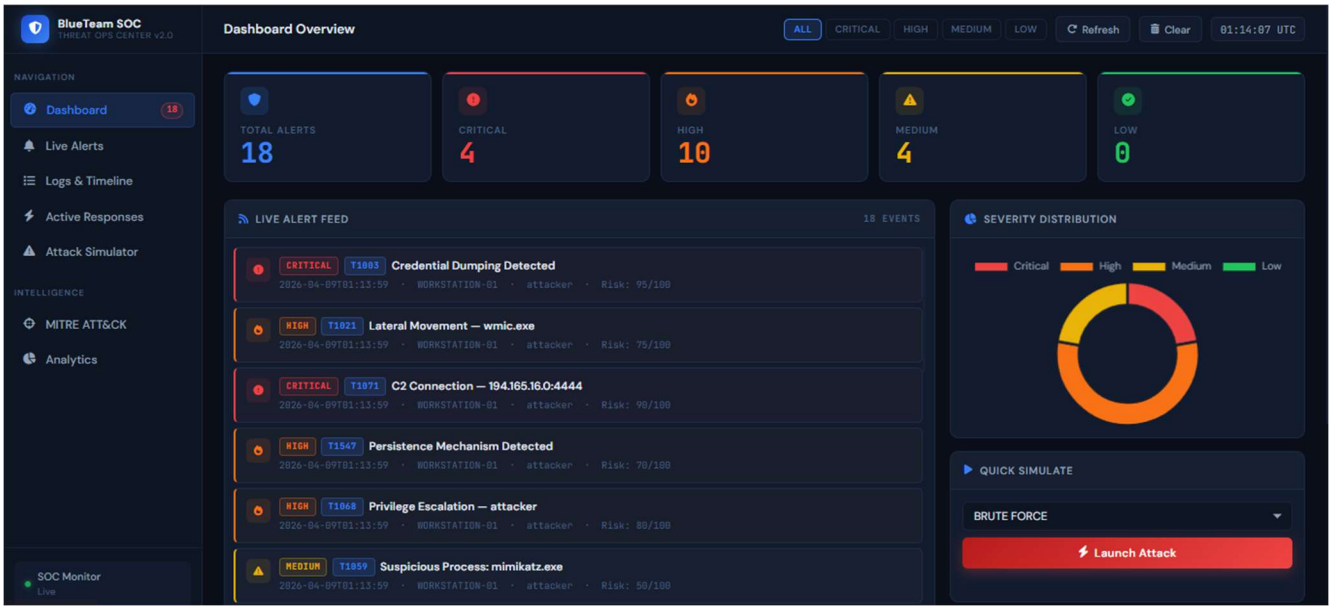

Figure 6: Output page of Unified blueteam threat detection and alert system

DISCUSSION

The evaluation results demonstrate that combining rule-based detection with risk correlation and automated response leads to better performance than traditional methods. Instead of relying on individual alerts, the system analyzes multiple related events together, which improves detection accuracy and reduces unnecessary alerts.

For example, a single suspicious PowerShell command may not always indicate an attack. However, when combined with registry modifications and unusual network activity, it becomes a strong indicator of malicious behavior. The risk scoring mechanism effectively captures this relationship and helps in identifying serious threats.

Another important advantage of the system is its automated response capability. Traditional SOC systems often require manual approval before taking action, which delays threat containment. In contrast, this system immediately executes predefined actions for high-risk events, significantly reducing response time.

However, the system has some limitations. It relies on predefined rules, which may not adapt to new or unknown attack patterns. Additionally, storing data in JSON files may not be efficient for handling very large volumes of data. Future improvements can address these challenges. Furthermore, the system demonstrates strong scalability due to its modular design, allowing new detection rules to be added without affecting existing components. The integration of real-time visualization enhances situational awareness for security analysts, enabling quicker understanding of ongoing threats. The use of lightweight technologies ensures that the system can be deployed even in resource-constrained environments. Another key observation is that automated response actions reduce dependency on human intervention, improving overall efficiency. The system also provides flexibility in configuration, allowing organizations to adjust thresholds based on their security requirements. In addition, the simulation module proves useful for training and testing purposes without impacting real systems. Overall, the proposed approach offers a balanced combination of accuracy, speed, and usability in modern SOC environments.

ISSN: **3107-6513**

CONCLUSION AND FUTURE WORK

This work presents the design and implementation of the Unified BlueTeam Threat Detection and Alert System, a complete SOC solution developed using Python. The system integrates log monitoring, rule-based detection, risk scoring, automated response, and a real-time dashboard into a single platform.

Experimental results show that the system achieves high detection accuracy while significantly reducing false alerts. It also provides fast response times, making it suitable for real-time cybersecurity applications. The system can be deployed independently or integrated with enterprise tools.

Future enhancements can further improve the system. Adding machine learning techniques such as anomaly detection can help identify unknown threats. Integration with external threat intelligence sources can provide additional context for analysis. Replacing JSON storage with database systems can improve scalability. Expanding the system to support cloud environments will also increase its applicability.

REFERENCES

1. Hofbauer, Jenny, and Kevin Mayer. "Blue Team Fundamentals: Roles and Tools in a Security Operations Center." *SECURWARE 2024: The Eighteenth International Conference on Emerging Security Information, Systems and Technologies*. IARIA, 2024.
2. Loizou, Pavlos. "Incident response case studies with Blue Team Labs Online." (2025).
3. Liu, Xiaoqun, et al. "Benchmarking llms in an embodied environment for blue team threat hunting." *arXiv preprint arXiv:2505.11901* (2025).
4. Sehgal, Kunal, and Nikolaos Thymianis. *Cybersecurity Blue Team Strategies: Uncover the secrets of blue teams to combat cyber threats in your organization*. Packt Publishing Ltd, 2023.
5. McLaughlin, Kevin Lynn. "Defense is the best offense: The evolving role of cybersecurity blue teams and the impact of soar technologies." *Edpacs* 67.6 (2023): 35-41.Pertuz, S., Puig, D., & Garcia, M. A. (2013). Analysis of Focus Measure Operators for Shape-from-Focus. Pattern Recognition, 46(5), 1415–1432. DOI: 10.1016/j.patcog.2012.11.011
6. Roddie, Megan, Jason Deyalsingh, and Gary J. Katz. *Practical Threat Detection Engineering*. 2023.
7. Westman, Niklas. "Testing an Alert Generation and Detection Framework: On Windows-based Systems and from a Threat Hunting Perspective." (2024).
8. Antwi, Noble Worlanyo. "Threat Detection in Multi-Cloud Environments." *Ensuring Secure and Ethical STM Research in the AI Era*. IGI Global Scientific Publishing, 2025. 111-190.
9. Mennuni, Massimo. *An Analysis of SOC Monitoring Systems*. Diss. Politecnico di Torino, 2023.
10. Hookana, Risto. "SOCs as Enablers for Continuous Threat Exposure Management." (2024).
11. Jones Sr, Joseph Anthony. *Cybersecurity methods to increase visibility, threat detection, and cyber defense in critical infrastructure*. Diss. Capitol Technology University, 2024.
12. Mohsin, Ahmad, et al. "A unified framework for human ai collaboration in security operations centers with trusted autonomy." *arXiv preprint arXiv:2505.23397* (2025).
13. Adesola, Adeyemi Afolayan. "Preparing for Unknown Cyber Threats: A Comprehensive Review of Framework for Speculative Threat Intelligence Using Cross-Domain Indicators." (2025).
14. Soman, Sruthi, and Zoheir Ezziane. "Leveraging Tech Towards Keeping Ahead of Cyber Threats and Alleviating Security Analysts' Alert Fatigue." *Proceedings of Eighth International Conference on Information System Design and Intelligent Applications*. Singapore: Springer Nature Singapore, 2024.
15. Westman, Niklas. "Testing an Alert Generation and Detection Framework: On Windows-based Systems and from a Threat Hunting Perspective." (2024).