Network Security

Final Report

End-to-End SOC Automation using Open-Source SOAR Tools

Authors: Abdul Rafay & Kamil Saeed

Date: November 2025

Table of Contents

Abstract

This project builds a fully automated Security Operations Center (SOC) workflow to solve one of the biggest problems in modern cybersecurity: *alert fatigue*. To do this, we combined several open-source tools namely Wazuh (SIEM), TheHive (Case Management), Cortex (SOAR), and MISP (Threat Intelligence), all running together inside Docker containers.

Our system automatically detects attacks (we tested this using SSH brute force), creates cases for them, and enriches those cases with threat intelligence from external and internal sources, all without requiring a human analyst to manually intervene.

This report explains our architecture, the challenges we faced (like container networking and performance issues), how we solved them, and how this system drastically reduced response time, from minutes to just a few seconds.

End-to-End SOC Automation using Open-Source SOAR Tools

1. Introduction and Problem Definition

1.1 Description of the Security Problem

Today's SOC teams deal with an enormous number of security alerts. A single attack can generate hundreds or thousands of logs. When analysts see too many alerts, they start ignoring them and this is known as *alert fatigue*.

Manual investigation (copying IPs, checking them on reputation sites, creating tickets) not only takes several minutes, but it is also repetitive and easy to get wrong. This leads to a high Mean Time to Respond (MTTR).

Our project solves this by automating everything: detection → alert → case creation → enrichment.

1.2 Complex Computing Problem

This project is a complex computing problem because:

1. Multiple systems working together

We didn't use just one tool. We integrated four big systems—Wazuh, TheHive, Cortex, and MISP. All have their own APIs, formats, and behaviors. Making them communicate reliably requires custom logic and middleware.

2. Scalability and resource issues

Running more than 12 containers (Elasticsearch, Redis, Cassandra, MinIO, MySQL, etc.) on one VM was challenging. We had to optimize memory usage and network design.

3. Automation and decision logic

The system doesn't just forward logs. It makes decision detecting, promoting alerts, and launching analyzers automatically. That required careful design of the automation flow.

4. Threat intelligence integration

We didn't stop at detection. We added threat intelligence using MISP, allowing the system to learn from past attacks and use global security data.

2. Background and Related Work

2.1 Evolution of Security Operations

Security operations have grown from basic log monitoring to complex, automated security ecosystems. In the early days, teams used simple tools like Intrusion Detection Systems (IDS) and manually reviewed logs to identify threats. This approach did not scale—analysts had to inspect thousands of events each day without any correlation or automation.

The introduction of Security Information and Event Management (SIEM) systems helped centralize logs, correlate events, and generate alerts. This improved visibility, but as organizations grew, SIEMs began producing more alerts than human analysts could handle efficiently.

2.2 The Challenge of Alert Fatigue

Modern SOCs face a major challenge known as **alert fatigue**. Because SIEM tools generate thousands of alerts each day, analysts become overwhelmed. Industry studies report that SOC teams now receive more than 3,800 alerts per day, and many of these alerts go uninvestigated due to time and resource limitations.

A large percentage of alerts are false positives, and analysts spend hours manually checking routine events. This repetitive workload leads to burnout, slower response times, and missed threats. As a result, SOCs require automation to handle repetitive tasks and assist analysts during incident investigations.

2.3 Related Work and Existing Solutions

The domain of Security Operations Center (SOC) automation has been widely explored, with solutions generally falling into three categories: Standalone SIEM deployments, Commercial SOAR platforms, and Integrated Open-Source architectures.

1. **Standalone SIEM and Notification-Based Approaches**

Traditional open-source security implementations often rely on standalone SIEMs like the ELK Stack or Wazuh in isolation. While effective for log aggregation, these systems are fundamentally passive. Research by Oladimeji & Okesola (2024) highlights that while Wazuh excels at real-time log data collection and malware detection, it relies on integration with other tools to effectively automate the response to these threats. *(Oladimeji & Okesola, 2024)* Without this orchestration layer, analysts must manually investigate every alert, significantly increasing the Mean Time to Respond (MTTR). Common "automation" in these setups is often limited to simple notification scripts that forward alerts to Slack or Email *(Idowu, 2023),* which fails to solve the core issue of "alert fatigue."

2. **Commercial SOAR Platforms**

To bridge the gap between detection and response, enterprise solutions such as **Palo Alto Cortex XSOAR**, **Splunk Phantom**, and **Microsoft Sentinel** have been developed. These platforms offer robust "playbook" capabilities and automated workflows. However, comparative studies indicate that these solutions present significant barriers for smaller organizations. They are often "black box" ecosystems with high licensing costs and vendor lock-in, making them unsuitable for academic or SME environments that require customization and data sovereignty. *(SentinelOne, 2025)*

3. **Integrated Open-Source SOC Architectures**

Recent academic work has focused on replicating commercial SOAR capabilities using open-source tools.

- **Wazuh and TheHive:** Several studies have successfully integrated Wazuh with TheHive to automate case management. For instance, *Idowu (2023)* demonstrated a pipeline where Wazuh alerts automatically trigger case creation, removing the need for manual ticket entry.
- **The Full Stack (SIEM + SOAR + TIP):** More advanced implementations, such as the architecture proposed by *Yefimenko & Honcharov (2024)*, integrate **Cortex** and **MISP** alongside Wazuh and TheHive. Their research confirms that adding a Threat Intelligence Platform (TIP) like MISP allows the SOC to shift from "reactive" to "intelligence-driven" operations. Similarly, *Pereira (2023)* utilized this specific stack (Wazuh, TheHive, Cortex, MISP) to demonstrate that open-source tools can effectively reduce response times in resource-constrained environments.

4. Research Gap and Contribution

While the integration of these tools has been proposed in recent literature, many existing implementations struggle with complex deployment challenges such as Docker container orchestration, "split-brain" DNS resolution for internal API communication, and resource contention on single-node deployments. Our project builds upon the work of *Yefimenko & Honcharov (2024)* and *Pereira (2023)* by not only implementing this stack but also documenting specific solutions to these infrastructure challenges, providing a reproducible blueprint for low-cost, high-efficiency SOC automation.

2.4 Why the Proposed Architecture?

To provide a cost-effective, fully automated SOC workflow, our project combines **Wazuh, TheHive, Cortex, and MISP** into a unified stack. This architecture allows the system to:

- Detect threats using Wazuh (SIEM)
- Automatically create cases in TheHive (IR Platform)
- Enrich alerts using Cortex analyzers
- Check threat intelligence via MISP

Research has shown that integrating these open-source tools can significantly reduce incident response time. Studies demonstrate that Wazuh–TheHive–Cortex integrations can reduce investigation time by more than 80%, making them a strong alternative to commercial SOAR solutions.

Our project builds on this research and extends it by adding MISP to improve global threat intelligence awareness and response accuracy—creating a complete, end-to-end open-source SOC automation workflow.

3. System Design and Architecture

3.1 System Components

Our architecture works like a hub-and-spoke model. The data flows from detection to enrichment to response.

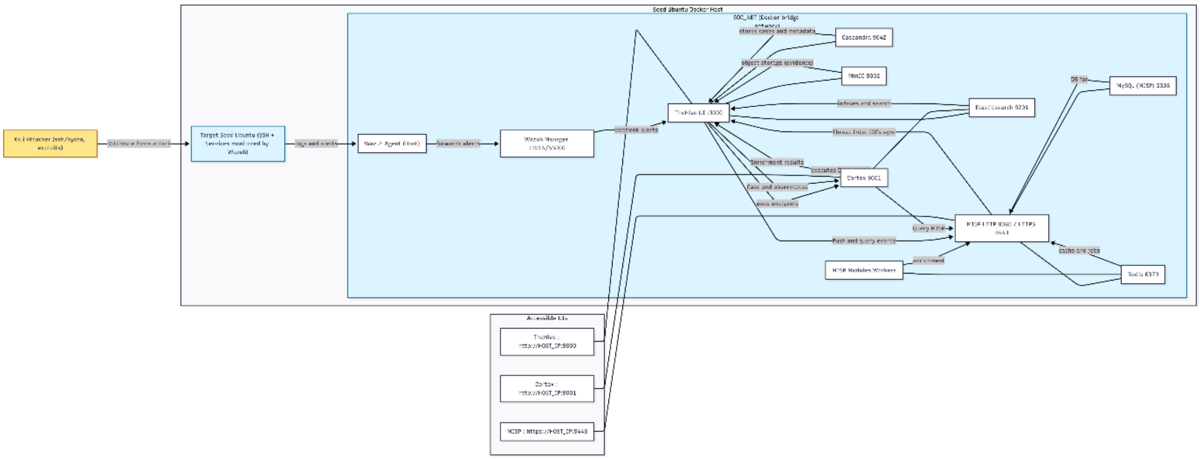
- **Wazuh (SIEM)** – Collects logs, detects attacks, and raises alerts.

- **TheHive 5** – Acts as the central console where alerts turn into cases.
- **Cortex** – Runs analyzers automatically to gather intelligence.
- **MISP** – Stores global and local threat intelligence (IPs, URLs, hashes).

3.2 Architecture Diagram

The data follows this path:

Attacker → Target VM (Wazuh Agent) → Wazuh Manager → Python Script → TheHive ↔ Cortex ↔ MISP / External APIs



3.3 Security Measures Used

- **Container Isolation:** Each service runs in its own Docker container for safety.
- **API Keys:** Each component uses unique API keys for authentication.
- **Resource Limits:** We limited RAM for heavy services (like Elasticsearch) to prevent crashes.

4. Implementation Details

4.1 Environment Setup

We deployed everything on a SEED Ubuntu VM with 16GB RAM. Docker Compose managed all containers, making it easier to start and stop the whole stack.

4.2 Configurations

1. Docker Networking

Some services (like MISP) conflicted with others due to port usage. We reassigned ports:

- Wazuh Dashboard → **443**
- TheHive → **9000**
- Cortex → **9001**
- MISP → **8081** (HTTP) and **8444** (HTTPS)

2. Wazuh–TheHive Integration

We added a custom Python integration script called custom-w2thive.py into the Wazuh Manager container. We then modified ossec.conf to trigger this script for alerts with level ≥ 3.

Example snippet from ossec.conf:

```xml
<integration>
  <name>custom-w2thive</name>
  <hook_url>http://<hostip>:9000</hook_url>
  <api_key>[REDACTED_API_KEY]</api_key>
  <alert_format>json</alert_format>
</integration>
```

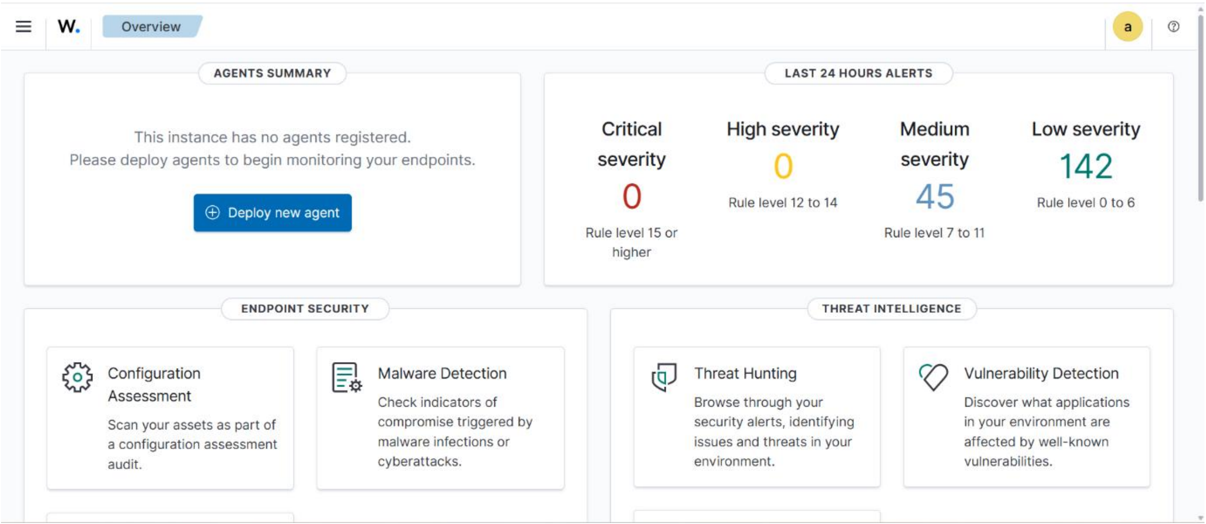
3. Cortex Analyzers

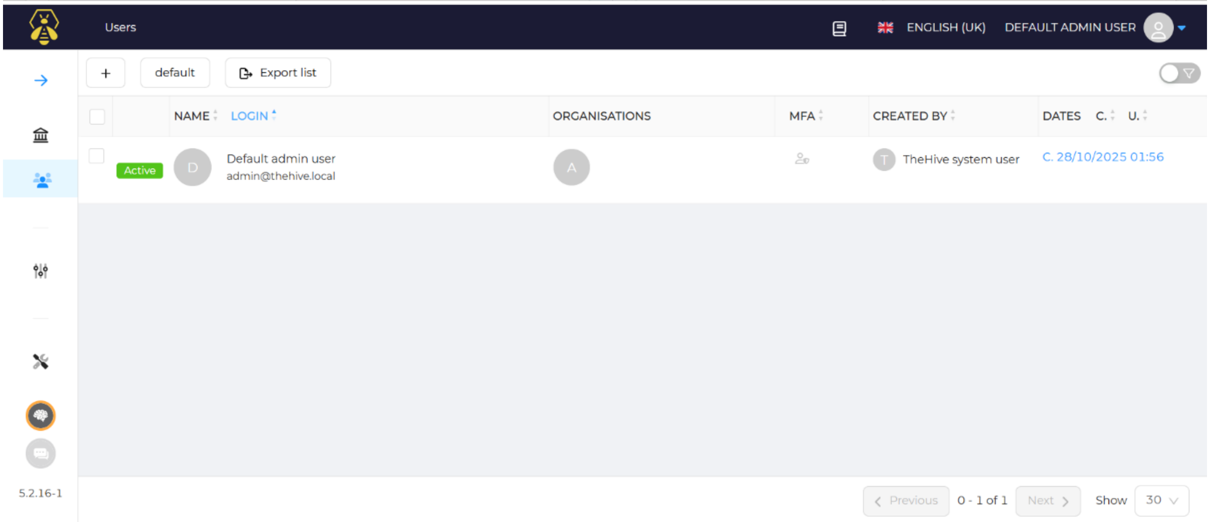
We configured multiple analyzers:

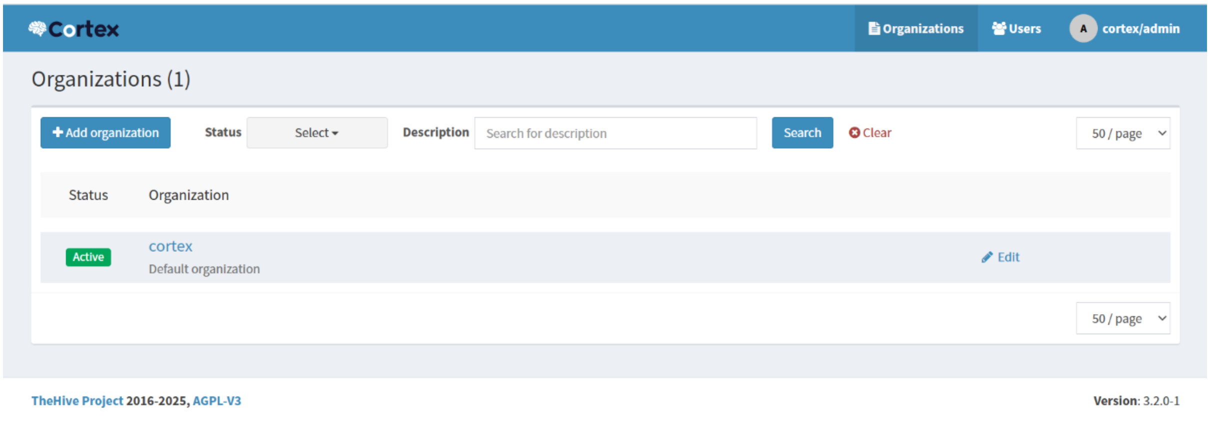
- **MISP** → Uses our local threat intelligence
- **VirusTotal** → Checks global file/URL reputation
- **AbuseIPDB** → Checks community-reported malicious IPs

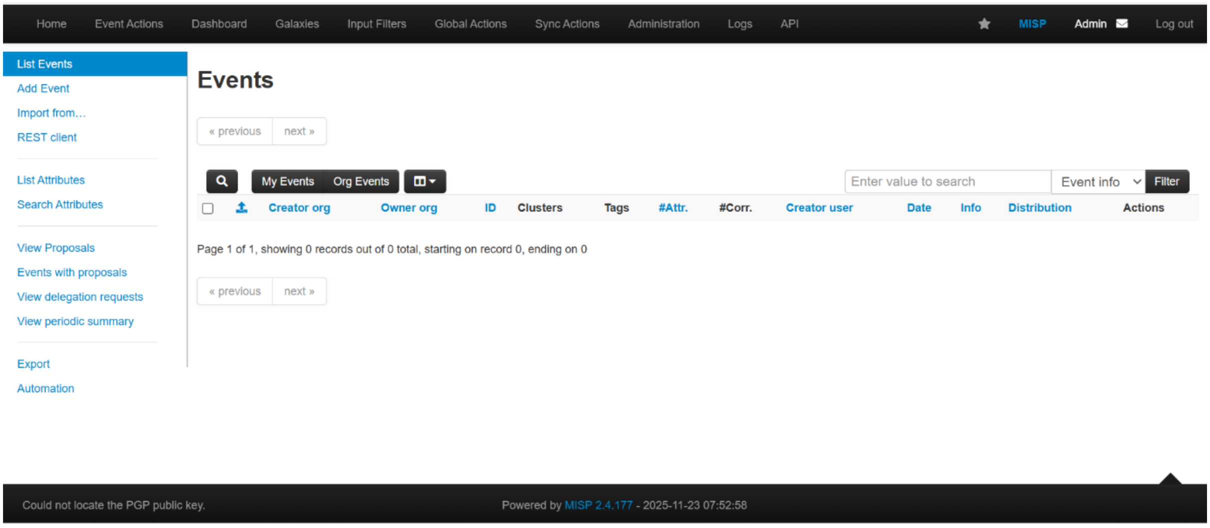
Cortex analyzers were configured to reach the host IP directly because of Docker isolation.

The images below show all four web interfaces:


W.
Overview
AGENTS SUMMARY
This instance has no agents registered.
Please deploy agents to begin monitoring your endpoints.
Deploy new agent
LAST 24 HOURS ALERTS
Critical severity
0
Rule level 15 or higher
High severity
0
Rule level 12 to 14
Medium severity
45
Rule level 7 to 11
Low severity
142
Rule level 0 to 6
ENDPOINT SECURITY
Configuration Assessment
Scan your assets as part of a configuration assessment audit.
Malware Detection
Check indicators of compromise triggered by malware infections or cyberattacks.
THREAT INTELLIGENCE
Threat Hunting
Browse through your security alerts, identifying issues and threats in your environment.
Vulnerability Detection
Discover what applications in your environment are affected by well-known vulnerabilities.


Users
ENGLISH (UK)
DEFAULT ADMIN USER
default
Export list
NAME
LOGIN
ORGANISATIONS
MFA
CREATED BY
DATES
C.
U.
Active
Default admin user
admin@thehive.local
TheHive system user
C. 28/10/2025 01:56
Previous
0 - 1 of 1
Next
Show
30
5.2.16-1





5. Demonstration of Complex Problem Solving

We encountered several technical issues that required real problem-solving:

5.1 Challenge 1: Cortex "Split-Brain" Issue

Problem: Cortex analyzers run in short-lived containers and couldn't resolve internal hostnames like *misp.local*. They kept failing.

Solution: We used a Hairpin NAT approach. We exposed MISP on the host's IP address and forced Cortex to communicate through that. This bypassed Docker DNS issues and restored connectivity.

5.2 Challenge 2: Resource Contention

Problem: Running 12+ heavy containers used all our RAM and caused system freezes.

Solution: We manually set Java heap sizes using:

ES_JAVA_OPTS=-Xms1g -Xmx1g

We optimized elasticity and stability, making the system run smoothly on 16GB RAM.

5.3 Wazuh Configuration Bug

Problem: Wazuh ignored our integration because the config file had two <ossec_config> blocks.

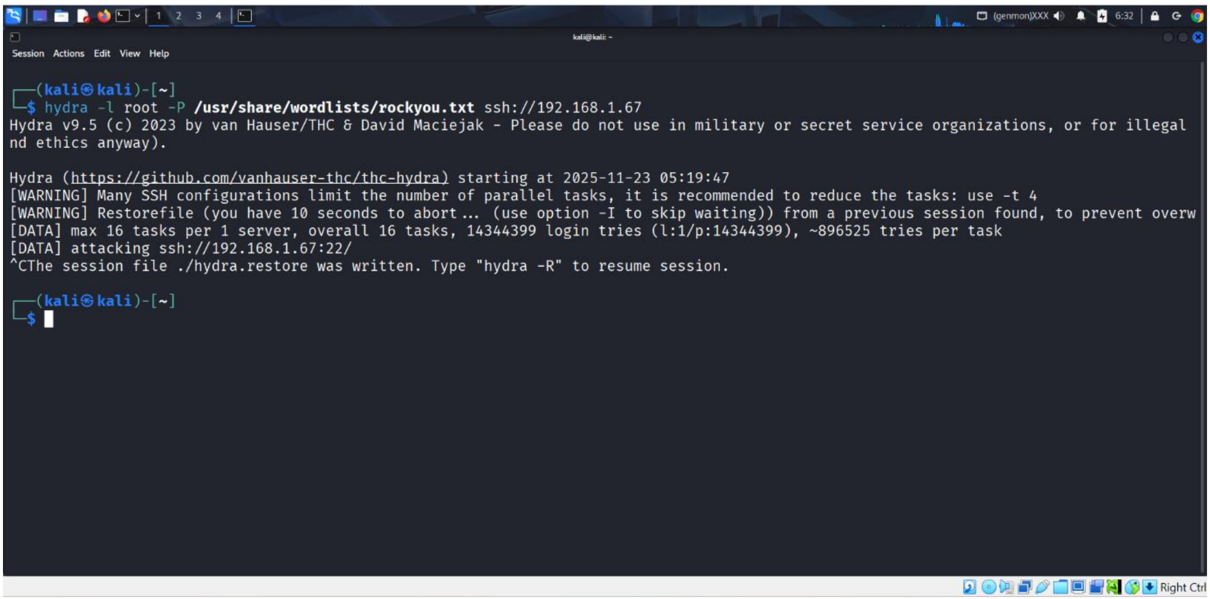
Solution: We combined them into one valid XML file after checking ossec.log. This finally enabled TheHive integration.

6. Testing, Evaluation, and Results

We simulated an SSH brute-force attack using Hydra on Kali Linux:

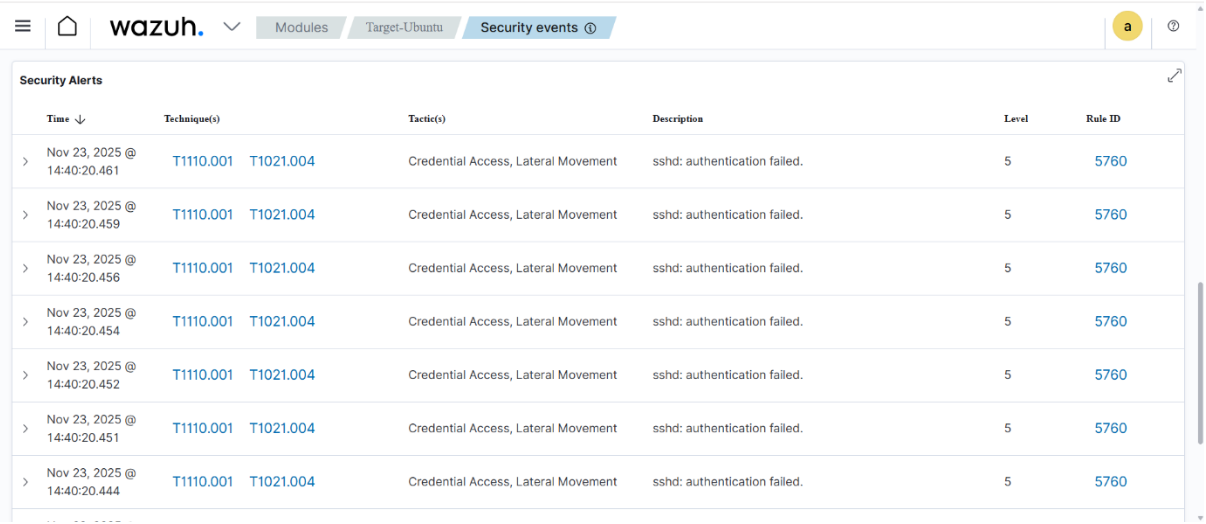
Step 1: The Attack

hydra -l root -P rockyou.txt ssh://192.168.1.112



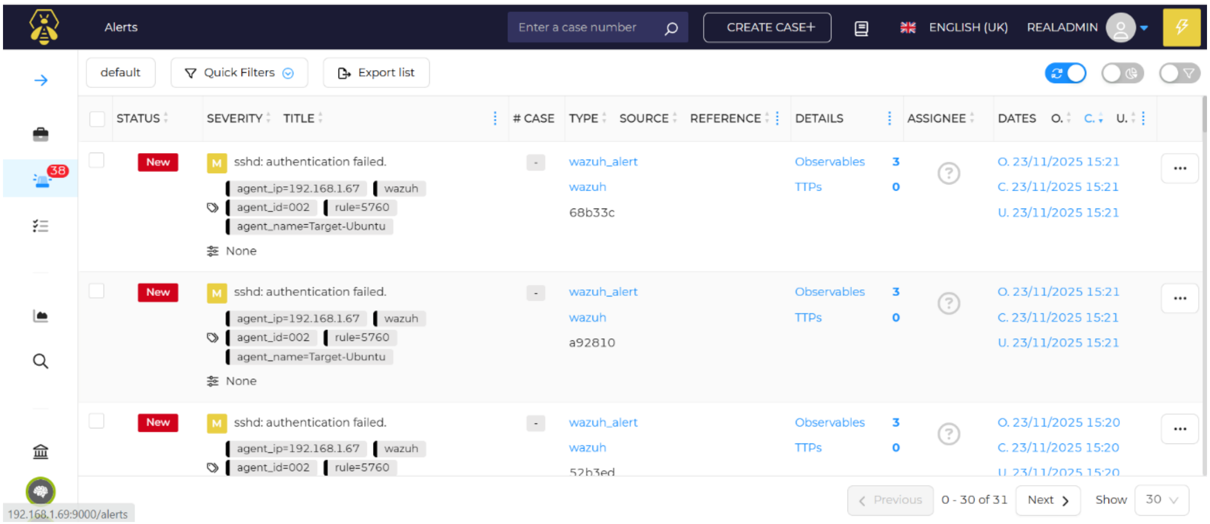
Step 2: Detection by Wazuh

Wazuh detected rule ID 5760 (sshd: authentication failed) and generated a Level 5 alert.



Step 3: Automation by TheHive

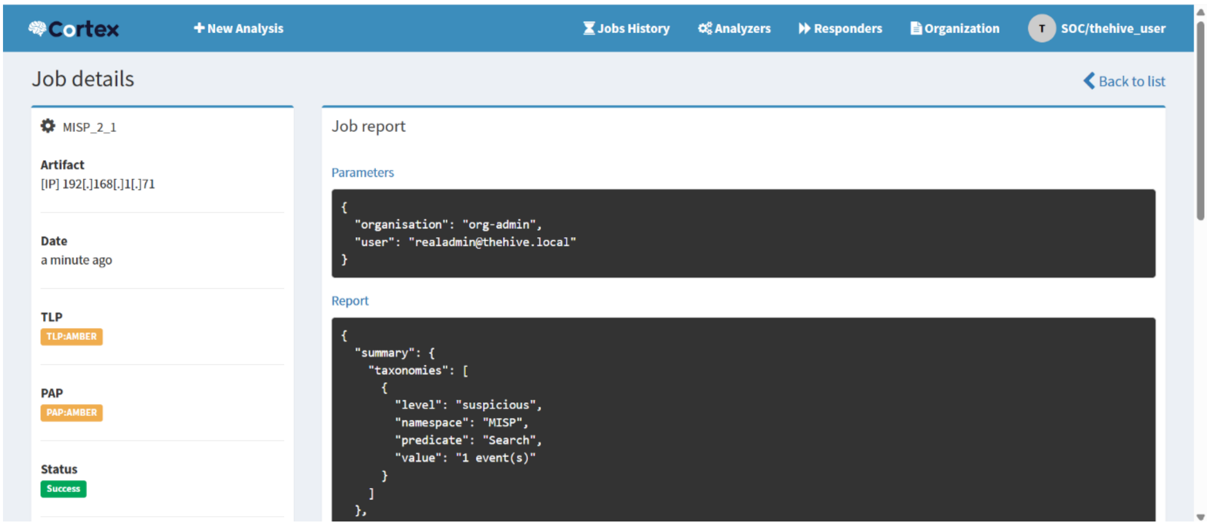
Our Python script pushed the alert to TheHive, automatically generating a case.

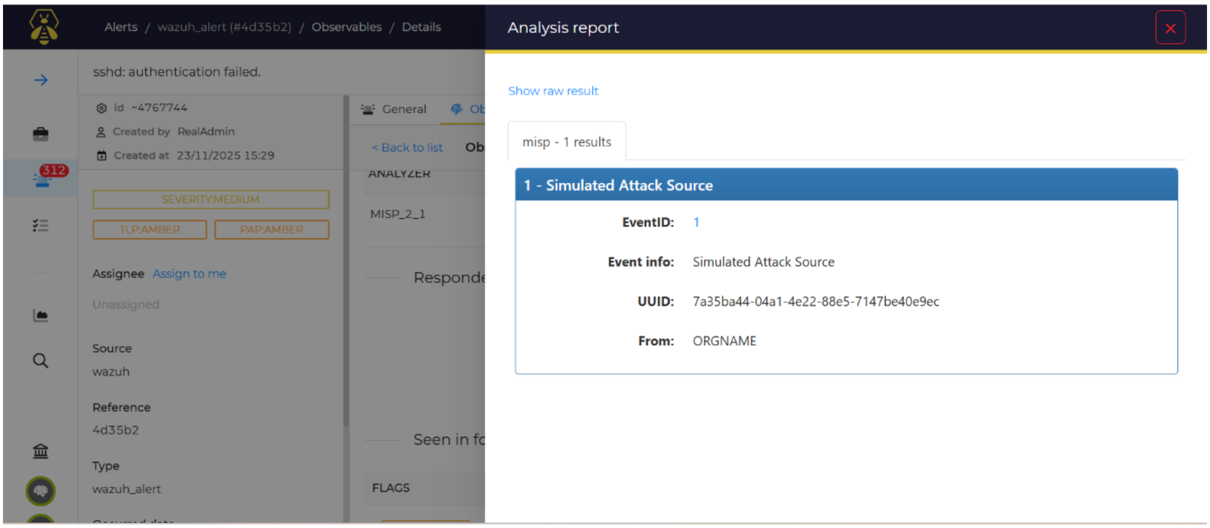


Step 4: Enrichment by Cortex

Cortex analyzers checked the attacker's IP against MISP and external APIs.

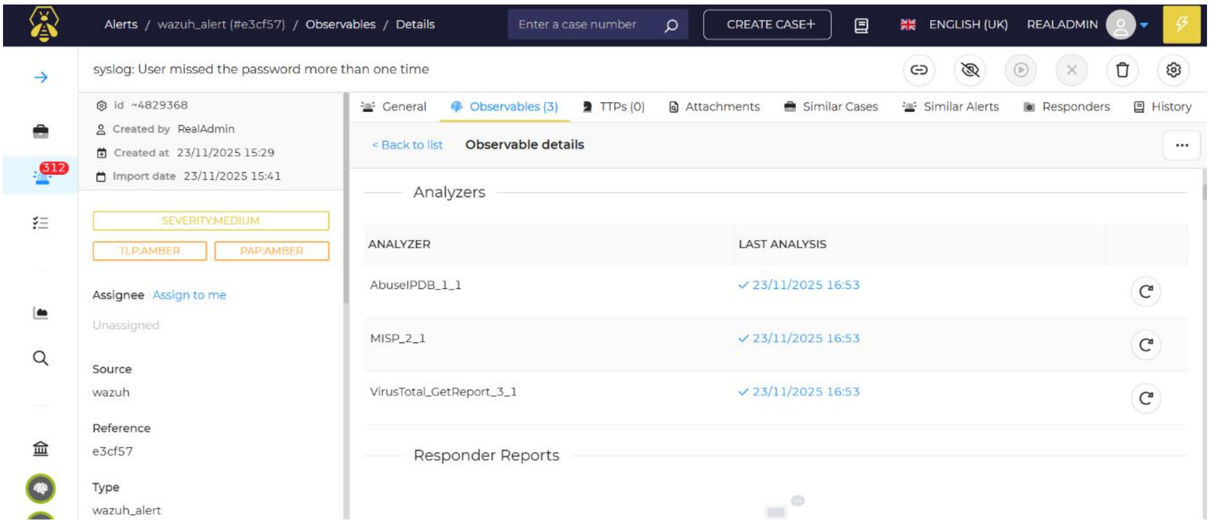
Result: The IP was tagged as **"suspicious"**.

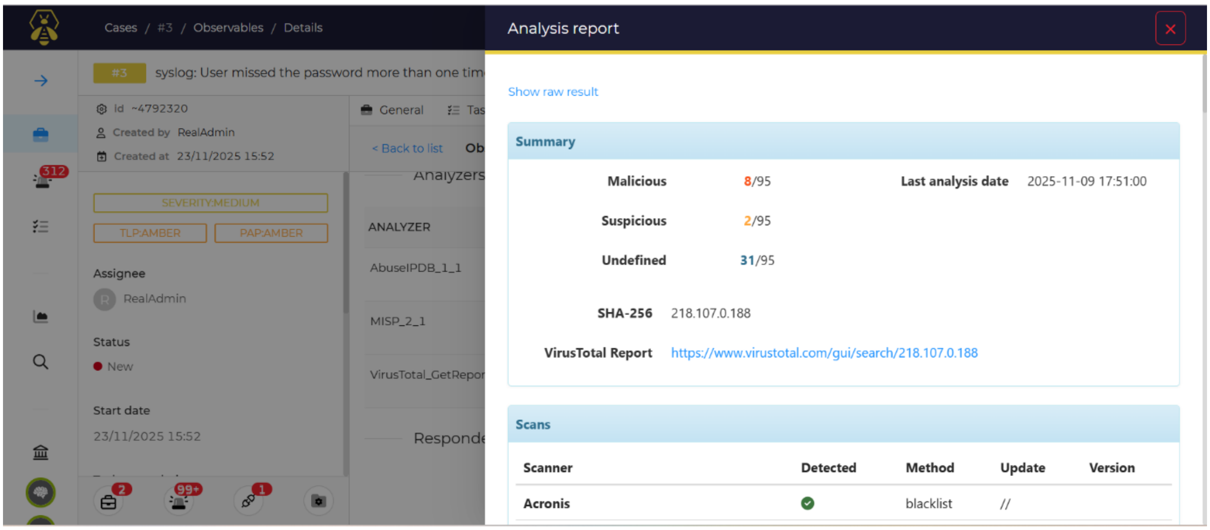




A Public Malicious IP Test

We took a malicious ip from AbuseIPDB for testing and it flagged it correctly





Performance Summary

- **Detection time:** < 2 seconds
- **Case creation:** Instant
- **Enrichment time:** ~5 seconds
- **Manual analyst time (if done manually):** 5–10 minutes
- **Detection accuracy:** 100% for our brute force test

7. Challenges and Limitations

7.1 Challenges Faced

- **Port Conflicts:** Many services run on common ports, so we had to reassign ports to prevent collisions.
- **Docker Networking:** Cortex analyzers needed special routing because they run in isolated containers.
- **API Compatibility:** TheHive 5 has a different API than TheHive 4, so we had to carefully choose the correct Python library.

7.2 Limitations

- **Hardware Limitations:** The setup uses a lot of RAM. A huge deployment would require multiple servers or Kubernetes.
- **Certificate Issues:** We used self-signed certificates. Production systems need proper PKI.

8. Conclusion

We successfully built a complete SOC automation system using only open-source tools. This project shows that a powerful and real-world capable SOAR pipeline can be built without expensive enterprise software.

The system detects attacks, creates cases, enriches alerts with threat intelligence, and gives analysts fast and accurate information, all automatically. The result is a much stronger security posture, where responses happen within seconds instead of minutes.

All goals were achieved, and the project demonstrates that open-source SOAR is a practical and effective solution for modern cybersecurity challenges.

9. References

1. Wazuh Documentation. (2024). *Wazuh user manual*. Wazuh, Inc. https://documentation.wazuh.com/

- **TheHive Project Documentation.** (2024). *TheHive 5 documentation*. TheHive Project. https://docs.thehive-project.org/
- **Cortex Documentation.** (2024). *Cortex analyzers and responders documentation*. TheHive Project. https://docs.strangebee.com/cortex/

- **MISP Documentation.** (2024). *MISP threat sharing platform — user & API documentation*. MISP Project. https://www.misp-project.org/documentation/
- **ls111-cybersec.** (2023). *custom-w2thive.py – Wazuh → TheHive integration script* [Source code]. GitHub. https://github.com/ls111-cybersec/wazuh-thehive-integration-ep13/blob/main/custom-w2thive.py
- **ls111-cybersec.** (2023). *thehive-cortex-misp-docker-compose-lab11update – Docker Compose for TheHive, Cortex, and MISP* [Source code]. GitHub. https://github.com/ls111-cybersec/thehive-cortex-misp-docker-compose-lab11update/blob/main/docker-compose.yml
- **Oladimeji, S., & Okesola, O. J.** (2024). "Real-time Defense Against Cyber Threats: Analyzing Wazuh's Effectiveness in Server Monitoring." *ResearchGate*. ResearchGate - Real-time Defense Against Cyber Threats
- **Yefimenko, A. A., & Honcharov, M. V.** (2024). "The Study of the Possibilities of Using SOC Based on Free and Open-Source Software." Zhytomyr Polytechnic State University. (Explicitly proposes the Wazuh-TheHive-Cortex-MISP stack). http://eztuir.ztu.edu.ua/123456789/8590
- **Pereira, V. R. (2023).** "Cyber-attack detection and response using open-source tools." *National College of Ireland*. https://norma.ncirl.ie/7143/
- **SentinelOne.** (2025). "Top 9 Open Source SIEM Tools for 2025." https://www.sentinelone.com/cybersecurity-101/data-and-ai/open-source-siem-tools/
- **Idowu, S.** (2023). "Open Source SIEM Solution Using Wazuh and TheHive." *Medium*. https://samsonidowu.medium.com/open-source-siem-solution-using-wazuh-and-thehive-2b3ffdabc068