أكاديمية كاوست
KAUST ACADEMY

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Insider Threat Simulation and Detection System

Abrar Khalid Alharbi, Ghaida Alqurashi, Shjoon Nasser Almutairi, Prof. Mudassar Aslam

Alharbiabrar598@gmail.com, ghaidaalqurashi@gmail.com, shjoon.ns.al@gmail.com, maslam@brookes.ac.uk

Date: August 24, 2026

Code: https://github.com/GhaidaAl-Qurashi/insider-threat-detection-system

Contents

1 Introduction 3

1.1 Problem Statement 3

1.2 Key Assumptions and Limitations 3

1.3 Summary of the Proposed System 3

2 Related Work 3

2.1 Insider Threat Taxonomies and Frameworks 4

2.1.1 Overview 4

2.1.2 Relevance to This Project 4

2.1.3 Strengths 4

2.1.4 Limitations / Gap This Work Addresses 4

2.2 Comparative Studies of SIEM Platforms 4

2.2.1 Overview 4

2.2.2 Relevance to This Project 5

2.2.3 Strengths 5

2.2.4 Limitations / Gap This Work Addresses 5

2.3 Summary and Positioning of This Project 5

2.4 Alert Fatigue in Security Operations Centres 5

2.4.1 Overview 5

2.4.2 Relevance to This Project 6

2.4.3 Strengths 6

2.4.4 Limitations / Gap This Work Addresses 6

2.5 The Human Element and Analyst Skill Gap in SOC Detection 6

2.5.1 Overview 6

2.5.2 Relevance to This Project 7

2.5.3 Strengths 7

2.5.4 Limitations / Gap This Work Addresses 7

3 Methodology 7

3.1 System Overview 7

3.2 Steps of System Operation 7

3.3 Solution Formal Formulation 8

3.3.1 Simulated Corporate Network Environment 9

3.3.2 External Attacker Environment 9

3.3.3 Wazuh Detection and Management Core 9

3.3.4 Risk Dashboard and Analyst Output 10

3.3.5 Custom Employee Risk Scoring Dashboard 11

3.4 Evaluation Procedure 13

3.4.1 Tools, Libraries, and Frameworks 13

3.4.2 Dataset and Test Input Creation 13

3.4.3 Metrics and Evaluation Criteria 13

4 Results 14

4.1 Component-level detection results 14

4.2 Attack Test Results 19

5 Discussion 21

6 Limitations and Future Work 22

7 Conclusion 23

1 Introduction

Insider threats represent one of the most difficult categories of cybersecurity incidents to detect, because those who are involved already have valid access to organizational systems, credentials, and data. Unlike external attackers who must first breach an organization's defenses, a malicious or negligent insider whether an unhappy employee, a careless staff member, or a compromised account operates from within the trust boundary of the organization. This makes insider activity difficult to differentiate from normal organization operations using traditional signature-based security tools.

Although insider events are less common than external attacks, industry reports regularly prove that they are more costly and take longer to identify and contain. This is mostly because of the fact that insider incidents can go undiscovered for a long time within normal user behavior. A successful insider attack, such as data exfiltration, espionage, or intellectual property theft, can have serious financial, reputational, and legal consequences, especially in industries that handle sensitive data. For this reason, simulating, identifying, and responding to insider threats is a crucial and useful skill set in Digital Forensics and Incident Response (DFIR).

1.1 Problem Statement

While almost all organizations make huge investments in external threat detection (firewalls, IDS/IPS, endpoint protection), they tend to not have advanced tools for identifying threats that come from authenticated, legitimate users. In order for security teams to create, test, and validate detection rules and investigative workflows before dealing with actual incidents, a controlled environment is required in which insider threat behaviors may be efficiently replicated, logged, and analyzed is required.

This addresses the following problem: How can realistic insider threat scenarios be simulated in a virtualized environment, and how well can a SIEM platform help to detect and investigate such threats using behavioral and log-based indicators?

1.2 Key Assumptions and Limitations

- A real organization environment's scale, network complexity, and background noise cannot be fully represented in the simulation environment, which is completely virtualized (VirtualBox).
- The project does not claim to cover every potential insider threat strategy; instead, detection is based on behavioral and log-based indicators that are visible to the SIEM (Wazuh).

1.3 Summary of the Proposed System

The proposed approach is a virtualized Insider Threat Simulation and Detection lab that runs a set of realistic threat scenarios against a monitored environment, including one involving a malicious insider and one involving an external attacker using stolen credentials. A centralized SIEM (Wazuh) collects, correlates, and alerts on log data produced by the simulated activity across the virtual network. The system demonstrates the full detection lifecycle, from behavioral simulation through log collection and correlation to alerting, risk scoring, and investigative analysis.

2 Related Work

This section reviews prior work in the areas that directly shaped the design of this lab: insider threat taxonomies and frameworks, comparative studies of SIEM platforms, alert fatigue in security operations centres, and the analyst skill gap in SOC detection.

2.1 Insider Threat Taxonomies and Frameworks

2.1.1 Overview

Two complementary frameworks inform how scenarios in this lab are classified. The CERT National Insider Threat Center at Carnegie Mellon University's Software Engineering Institute has maintained a corpus of real insider incident cases since 2001; the sixth edition of its Common Sense Guide to Mitigating Insider Threats, drawn from more than 1,500 documented cases, groups insider activity into five categories: theft of intellectual property, IT sabotage, fraud, espionage, and unintentional insider threat [1]. While CERT classifies insiders by motive, the MITRE ATT&CK framework classifies adversary behaviour by techniques, tactics, and procedures (TTPs) an actor employs against a system [2]. ATT&CK was designed primarily for external adversaries, but MITRE's Center for Threat-Informed Defense extended it specifically to insiders through the Insider Threat TTP Knowledge Base, built from real case data contributed by organizations including Microsoft, JPMorgan Chase, CrowdStrike, and Verizon [3].

2.1.2 Relevance to This Project

These two frameworks give the project a principled way to both select scenarios and verify detection. CERT's motive-based categories justify why a given scenario (e.g., an employee copying files before resignation) represents a real, well-documented threat category rather than an arbitrary exercise, while ATT&CK / Insider Threat TTP KB technique IDs give a way to report a scenario as "detected" or "not detected" against a recognized external standard rather than the team's own informal judgement.

2.1.3 Strengths

CERT's taxonomy is grounded in a very large real-world case base spanning two decades, giving it strong external credibility. MITRE's Insider Threat TTP Knowledge Base offers technique-level granularity and is up to date, the first cross-organizational, evidence-based knowledge base built specifically around insider rather than external behavior.

2.1.4 Limitations / Gap This Work Addresses

Both resources are classification frameworks, not implementations. They describe how to categorize a threat, but neither runs a detection pipeline or validates that a given SIEM configuration actually catches the behavior it describes. This project operationalizes both frameworks: each planned scenario is grounded in a CERT category and mapped to specific ATT&CK / Insider Threat TTP KB technique IDs, then actually executed on real endpoints and checked against real Wazuh alerts.

2.2 Comparative Studies of SIEM Platforms

2.2.1 Overview

Manzoor et al. deployed Wazuh, Elastic Security, AlienVault OSSIM, and SIEMonster side by side in a representative small-enterprise test network and measured both events-per-second (EPS) throughput and a weighted feature score across ten essential and five desirable SIEM capabilities [4]. Wazuh achieved the highest EPS across all log sources tested and the highest overall feature score, ahead of Elastic Security, SIEMonster, and OSSIM. Splunk's free tier was excluded from the detailed evaluation because its daily ingestion cap makes it unsuitable for anything beyond a toy deployment.

2.2.2 Relevance to This Project

This is a direct, independent justification for the project's choice of Wazuh over Splunk or a bare Elastic Stack.

2.2.3 Strengths

Unlike vendor comparison pages or blog posts, this is a peer-reviewed, testbed-based study with a reproducible, quantitative scoring methodology across fifteen named criteria, rather than a subjective or marketing-driven comparison.

2.2.4 Limitations / Gap This Work Addresses

The study's test scenarios are generic network and endpoint attacks (DoS floods, file-integrity-monitoring changes) rather than insider threat behavior specifically. This project narrows that gap by evaluating Wazuh's detection coverage against a curated set of realistic insider threat scenarios rather than generic network attacks.

2.3 Summary and Positioning of This Project

Relative to the prior work reviewed above, our project differs in three ways:

- **Live telemetry over synthetic data.** Unlike the CERT dataset generator [5], the "malicious insider" and "normal employee" activity in this lab is produced by real actions on real Windows 11 endpoints, captured by an actual Wazuh agent and correlated by an actual Wazuh manager, end-to-end, not simulated at the log-file level.
- **A framework-grounded scenario set.** The provisionally selected scenarios, IP theft/data exfiltration, unauthorized or off-hours access, privilege escalation, IT sabotage, and credential misuse, already align with this taxonomy: intellectual property theft and IT sabotage map directly onto two of CERT's five primary categories [1], while privilege escalation and credential misuse correspond to well-documented ATT&CK techniques such as Valid Accounts (T1078) and Abuse Elevation Control Mechanism (T1548) that also appear in the Insider Threat TTP Knowledge Base [2, 3].
- **An evidence-based SIEM choice.** The selection of Wazuh is consistent with the only available head-to-head, testbed-based comparison of open-source SIEM platforms [4], rather than being based on cost or familiarity alone.

At the same time, this project inherits the limitation common to small lab-scale studies such as Catescu's [6]: with two employee endpoints and a training-scale time window, it cannot claim the network complexity, user population, or background noise of a real organization. Its contribution is therefore best read as a reproducible, framework-grounded demonstration of the full insider threat detection lifecycle: simulate, collect, correlate, alert, investigate, rather than a claim of production-scale detection performance.

2.4 Alert Fatigue in Security Operations Centres

2.4.1 Overview

Security Operations Centres (SOCs) lean heavily on automated monitoring tools to process the massive volume of logs generated by modern enterprise networks. However, this heavy reliance introduces a well-documented side effect: alert fatigue. In their review, Tariq et al. [7] cite a Trend Micro survey finding that 54% of SOC teams feel overwhelmed by alert volume. A parallel Tines survey cited in the same work highlights widespread practitioner burnout alongside rising operational workloads.

Tariq et al. [7] categorize the root causes of alert fatigue into four primary factors:

1. Staff and skills shortages
2. High false-alarm rates
3. Disconnected and overloaded dashboards
4. Inefficient standard operating procedures (SOPs)

The literature further details a fundamental trade-off in automation-based mitigation: recall-focused strategies designed to capture rare, high-severity threats routinely flood analysts with low-confidence alerts, whereas precision-focused strategies designed to suppress noise risk letting subtle, low-frequency attacks slip through undetected [7].

2.4.2 Relevance to This Project

This project directly incorporates the insights from Tariq et al. [7] to proactively bypass the high false-alarm rates that plague typical enterprise deployments. Because default Wazuh rule sets lack organizational role context and department-specific access boundaries, relying solely on out-of-the-box detection inevitably generates massive false-positive noise.

To address this structural flaw, this project uses pre-script and custom correlation rules tailored across multiple insider threat behaviors including privilege abuse, anomalous off-hours access, and mass file exfiltration. By baking contextual role boundaries and fine-tuned, adjustable thresholds into rule development from the start, the proposed architecture intentionally solves the recall–precision tension identified in the literature, avoiding the alert floods and analyst burnout that burden conventional SOCs.

2.4.3 Strengths

Grounded in empirical survey data and literature reviews, Tariq et al. [7] establish that alert fatigue and false-positive churn are systemic structural challenges across modern security operations rather than isolated configuration failures. Referencing this research gives the project a literature-backed baseline for evaluating detection performance, ensuring that rule tuning accounts for operational analyst capacity alongside traditional detection rates.

2.4.4 Limitations / Gap This Work Addresses

While literature extensively documents alert fatigue and the mathematical trade-off between precision and recall, academic studies frequently examine these dynamics in abstract or survey-based contexts without demonstrating how specific SIEM engines handle them under custom rule sets. This project operationalizes these theoretical insights: by deploying hand-written, threshold-based correlation rules in Wazuh for targeted insider scenarios, it directly measures, documents, and evaluates the real-world friction between rule sensitivity and analyst noise in a live environment.

2.5 The Human Element and Analyst Skill Gap in SOC Detection

2.5.1 Overview

The human element and the growing skill gap among SOC analysts are important but often overlooked factors in security operations. While automation and SIEM platforms provide crucial data visibility, effective threat detection especially against advanced insider threats still depends heavily on human intelligence, domain expertise, and contextual reasoning. Crumpler and Lewis [8] establish that the global cybersecurity workforce shortage is driven less by a quantitative lack of personnel and more by a qualitative gap in hands-on technical capability and critical soft skills:

employers report that cybersecurity graduates are often strong in high-level policy and compliance knowledge but lack practical skills such as log analysis, system internals, and real-time detection engineering, alongside soft skills like analytical problem-solving and cross-team communication.

2.5.2 Relevance to This Project

This documented skills gap has direct consequences for insider threat detection specifically: automated tools can flag anomalies, but they struggle to distinguish malicious intent from legitimate access when an insider is already operating within their authorized permissions. Closing this detection gap requires analysts capable of designing custom detection logic grounded in human behavioral baselines recognizing data-staging patterns, cross-departmental file browsing, or off-hours access rather than relying on generic automated signatures.

2.5.3 Strengths

The CSIS workforce-gap findings are grounded in employer-reported data rather than anecdote, giving this project an evidence base for why analyst-designed detection logic (rather than default rule sets alone) is necessary.

2.5.4 Limitations / Gap This Work Addresses

This project addresses precisely the gap identified by Crumpler and Lewis [8]: by pairing human-designed detection logic with SIEM customization (custom Wazuh rules and Active Response), it demonstrates how analyst-driven detection engineering can compensate for the limitations of automation alone.

3 Methodology

3.1 System Overview

This project designs and implements a virtualized Insider Threat Simulation and Detection Laboratory that replicates a small corporate network environment in which two realistic threat scenarios are executed, logged, and analysed end-to-end. The system is built around three functional zones illustrated in Figure 1: a simulated corporate endpoint layer hosting the insider persona and victim machines (Scenario 1), an isolated external attacker layer operating from a Kali Linux VM (Scenario 2), and a centralized Wazuh detection and management core that collects telemetry from all endpoints, applies built-in and custom detection rules, and surfaces alerts through an analyst risk dashboard.

Every behavioural action performed during scenario execution generates observable, real telemetry on real Windows 11 and Ubuntu endpoints, forwarded by Wazuh agents to the manager for correlation and alerting completing the full detection lifecycle from behavioural simulation through to analyst-facing output.

3.2 Steps of System Operation

Phase 1 - Baseline establishment:. Before any malicious activity is introduced, scripted normal-user behaviour is executed across both the Finance and HR workstations over a simulated one-week window. This generates a realistic behavioural baseline (logon times, typical file-access patterns, network activity) against which anomalies are later measured, and is used to tune the custom Wazuh detection rules to minimize false positives during the subsequent attack phases.

Phase 2 - Scenario execution:. The two threat scenarios are executed in sequence:

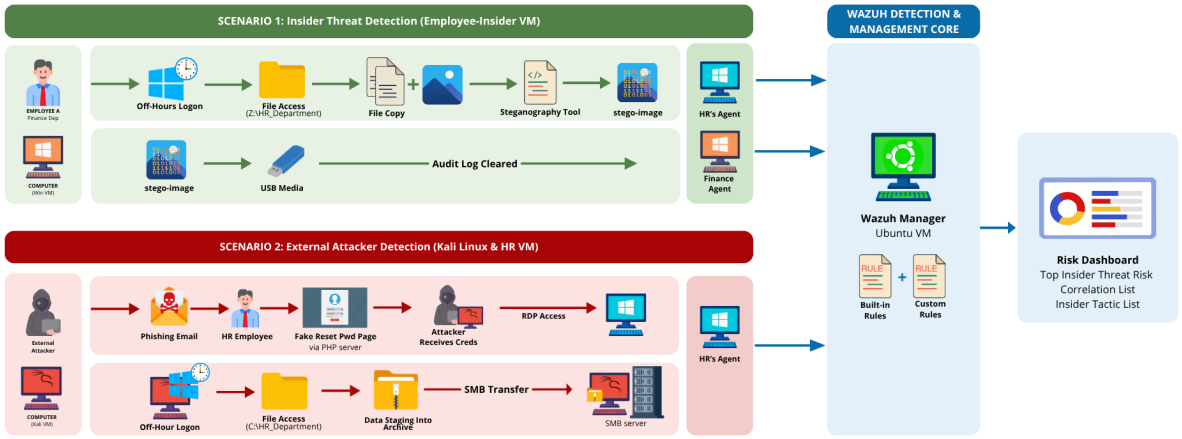


Figure 1 System Architecture

- **Scenario 1 - Insider threat:** performed from the Finance workstation (agent name: Employee-Insider) following a six-step attack chain — off-hours logon outside the 9am–5pm window, privilege abuse by accessing the HR Department shared folder, mass file copy of sensitive documents to a local staging directory, steganographic exfiltration by embedding staged files inside an innocuous image, USB/removable-media copy as a secondary exfiltration path, and log tampering by clearing Windows Event Logs to cover tracks.
- **Scenario 2 - External attacker:** performed from the Kali Linux VM against the HR workstation (agent name: Employee-Normal, whose credentials are compromised) following a seven-step chain credential harvesting via a fake Meridian Holdings password-reset page hosted on the attacker VM, remote authentication to the HR workstation over RDP using the phished credentials, off-hours access timing to reduce detection risk, browsing and reading sensitive HR/Executive files, data staging into a Zip archive, and data exfiltration to the attacker VM via SMB transfer using Impacket.

Phase 3 - Detection and analysis:. As each scenario step executes, Wazuh agents on the relevant endpoints forward telemetry to the manager in real time. The manager evaluates each incoming event against both default and custom rules and raises alerts where rules match. These alerts feed both the native Wazuh dashboard, used for raw event triage, and a custom-built Employee Risk Scoring dashboard (described in Section 3.3.5) designed specifically for insider threat investigation. This custom dashboard presents three analyst-facing views:

- **Top risk actors:** Ranks employees by cumulative alert severity score, surfacing which user account is generating the most anomalous activity across the monitored window, so analysts can immediately identify the highest-priority subject without manually reviewing individual alerts.
- **Log correlation:** Links related events across the attack timeline into a single coherent chain, so that individually low-severity events are surfaced together as a connected incident rather than appearing as isolated, unrelated alerts.
- **Tactic breakdown:** Maps each flagged behaviour to its corresponding MITRE ATT&CK technique, giving analysts a structured view of what kind of threat activity is occurring (e.g., credential access, data staging, exfiltration).

3.3 Solution Formal Formulation

3.3.1 Simulated Corporate Network Environment

The corporate environment consists of four virtual machines hosted in VirtualBox and connected over a Tailscale mesh network, collectively representing the internal infrastructure of the organization:

- **HR Workstation (Windows 11, agent: Employee-Normal)** serves as the primary victim machine in both scenarios. It hosts the sensitive HR and Executive file repository (`C:\02_Restricted_Executive_HR\Executive\`) which has the files that both the insider and the external attacker target. A Wazuh agent is installed and configured to ship Windows Event Log, Sysmon, and FIM telemetry to the manager.
- **Finance Workstation (Windows 11, agent: Employee-Insider)** represents the insider persona. This machine executes all six insider-threat behaviours in Scenario 1. A Wazuh agent captures process-creation, file-access, and logon events generated during execution.
- **Linux File Server (Ubuntu Server)** hosts the organization's shared drive via Samba. A Wazuh agent running auditd provides supplementary file-access, process-creation, and network-connections telemetry from this host.
- **Wazuh Manager (Ubuntu Server)** the centralized collection, correlation, alerting, and risk-scoring core.

3.3.2 External Attacker Environment

The external attacker is represented by a Kali Linux VM connected to the HR workstation through a deliberately isolated VirtualBox Internal Network segment (attacker-net, subnet 192.168.100.0/24). This segment is decoupled from the Tailscale-connected corporate network to accurately reflect the network boundary an external attacker would face in a real organization.

The Kali VM hosts two attacker-side components shown in Figure 1: a credential harvesting page (a fake Meridian Holdings SSO password-reset portal, served via PHP on port 80, designed to capture HR employee credentials) and an SMB exfiltration receiver (using Impacket's `smbserver.py`) used to receive the stolen archive after the remote session is established.

3.3.3 Wazuh Detection and Management Core

The Wazuh manager runs on a dedicated Ubuntu Server VM and serves as the centralized collection, correlation, and alerting engine, consisting of two rule layers:

- **Built-in rules** provide baseline coverage for generic security events such as logon success/-failure, FIM change notifications, log tampering, and Sysmon process-creation events.
- **Custom rules** the primary technical contribution of this project. Seven custom rules were written by the team to detect behaviours that Wazuh's default rule set does not cover, listed in Table 1.

Active Response is used for steganography detection: when FIM detects a new or modified image file in the monitored folder, it triggers a Python script that calculates the file's Shannon entropy and bytes-per-pixel ratio, returning a high-severity Wazuh alert if the value exceeds the defined threshold indicating potential hidden data embedded in the image.

Table 1 Custom Wazuh Detection Rules

Rule ID	Behaviour	The Detection Logic	MITRE ATT&CK mapping
Scenario 1			
100103	Off-hours logon	Time-window match on Event 4624 outside 09:00–17:00	No MITRE ID timing con- dition on a login event; not a distinct technique
100105	Mass file copy / staging	FIM event frequency exceeding 10 events within a 60-second window	T1074
100020	Steganographic exfiltration	Active Response trigger, entropy and Pixel-per-Byte check Python scripts that is alerted if they exceed threshold	T1027.003
100021	USB / removable media	Windows Plug-and-Play device-arrival event matching mass-storage device class	T1092
Scenario 2			
100026	RDP access RDP access	Possible remote access via compromised credentials	T1021.001
100027	Stolen Credentials authenticated On RDP Connection	100027 Stolen Credentials authenticated On RDP Connection Explicit-credential logon and activity attempted by The Attacker posed as the Employee	T1078
100028	SMB connection and data exfiltration	Outbound SMB connection permitted to attacker-net IP	T1048.002

Log tampering (Wazuh built-in rule 63103) and sensitive-file access / archive staging (built-in FIM rules 550, 553, 554, scoped to the Restricted HR directory) reuse Wazuh's default detection logic as-is in both scenarios, and off-hours access in Scenario 2 reuses rules 100103/100104 from Scenario 1.

3.3.4 Risk Dashboard and Analyst Output

All alerts generated by both default and custom Wazuh rules are forwarded in real time to a custom insider threat visualization dashboard linked directly to the Wazuh manager. Unlike the general-purpose Wazuh web UI, this dashboard is built specifically around the insider threat use case and presents data across three structured views:

- **Top Insider Threat Risk (risk actor ranking)** Aggregates all alerts by the source user account and computes a cumulative risk score based on alert severity and frequency. This view answers the analyst's first question, "Who should I be looking at?", without requiring manual triage of the full alert stream. In the context of this project, Insider employee ("A" Finance employee) and the compromised HR account (Scenario 2) both surface as high-risk actors as their respective scenarios progress.
- **Correlation List (timeline correlation)** Links related alerts from the same source account

across the full observation window into a single connected incident chain.

- **Insider Tactic List (MITRE ATT&CK mapping)** Displays each detected behaviour mapped to its corresponding MITRE ATT&CK technique ID, giving the analyst a structured breakdown of the threat activity. For example:
  - **Mass file copy:** T1074 (Data Staged)
  - **Steganography:** T1027.003 (Obfuscated Files or Information: Steganography)
  - **USB storage connection:** T1092 (Communication Through Removable Media)
  - **RDP access:** T1021.001 (Remote Desktop Protocol)
  - **Stolen credentials:** T1078 (Valid Accounts)
  - **SMB connection & data exfiltration:** T1048.002 (Exfiltration Over Alternative Protocol)

  This structured view allows analysts to quickly assess the sophistication and intent of an incident at a glance.

3.3.5 Custom Employee Risk Scoring Dashboard

Wazuh's own interface reports alerts individually; it has no built-in concept of a cumulative "how risky does this employee look right now" score across multiple behaviors over time. To close this gap, the team built a second, purpose-made layer on top of Wazuh:

- A Python script running on the Wazuh manager VM periodically reads new alerts (filtered by agent name and rule ID) and assigns each a weight based on its severity and behaviour type (e.g., off-hours access and USB connection are weighted lower than steganographic exfiltration or SMB exfiltration to an external IP).
- The script accumulates these weighted scores per employee account into a single running risk score and writes the result to one unified JSON file (a single handoff schema was chosen over multiple per-section files specifically to avoid coordination overhead between the scoring script and the dashboard front end).
- A lightweight, CORS-enabled Python HTTP server on the Wazuh manager VM serves this JSON file across the Tailscale network, and a cron job re-runs the scoring script on a fixed interval so the file stays current without manual copying between machines.
- The front-end dashboard (the "Employee Risk Scoring Interface") polls this JSON file automatically and renders it as a ranked list of employees by cumulative risk score, alongside the correlated alert timeline and ATT&CK tactic tags for the selected employee.

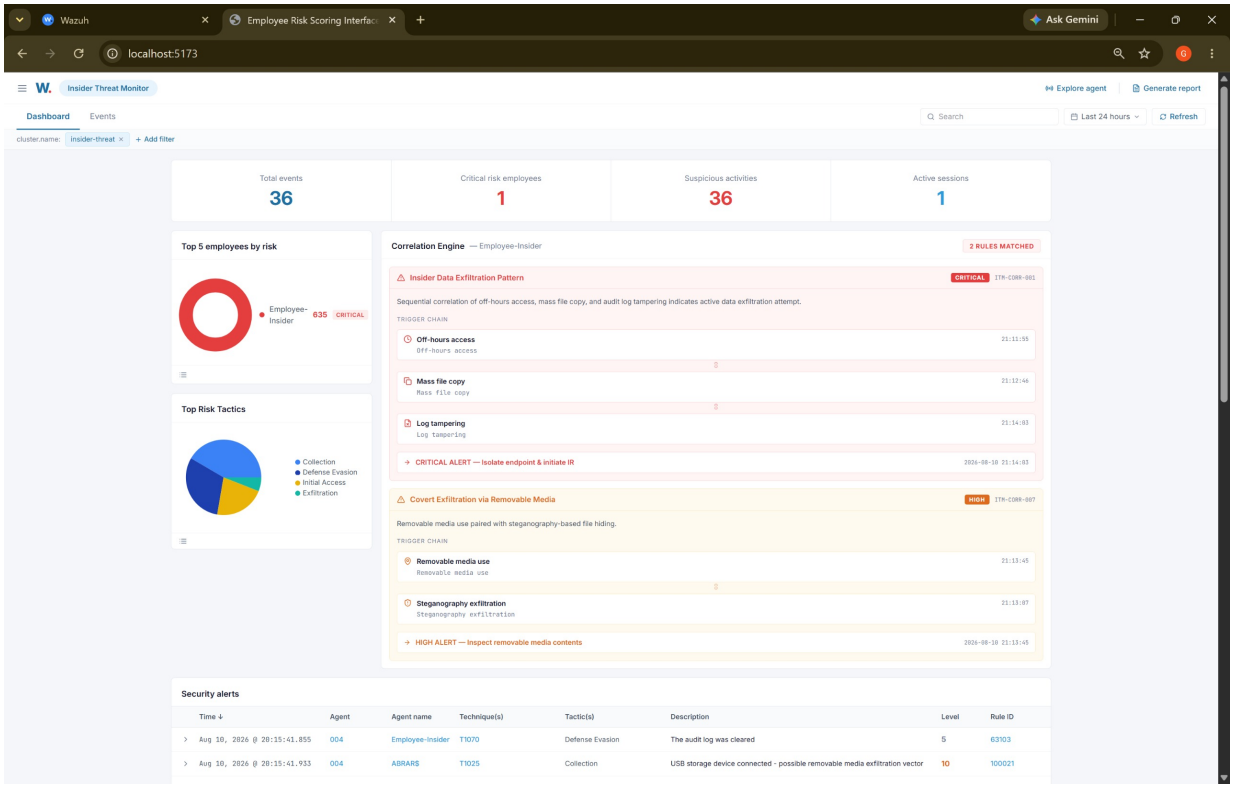


Figure 2 Risk Scoring Dashboard

3.4 Evaluation Procedure

3.4.1 Tools, Libraries, and Frameworks

Category	Tools / Frameworks
SIEM / HIDS	Wazuh Manager and Agent (v4.14)
Networking	Tailscale (mesh VPN across distributed team VMs)
Virtualization	VirtualBox (VM hosting, USB device passthrough)
Endpoint OS	Windows 11 (agent/persona VMs), Kali Linux (attacker VM)
Custom detection scripting	Python 3, Pillow (PIL), watchdog
Steganography tooling	OpenStego (payload embedding, used for both scenario construction and detection validation)
Attacker tooling	xfreerdp (remote access), 7-Zip (data staging/archives), certutil (LOLBin encode/decode),), Impacket smbserver.py (SMB exfiltration receiver)
Rule authoring/testing	nano (rule editing), wazuh-logtest (synthetic rule validation)
Threat mapping	MITRE ATT&CK framework (technique tagging per custom rule)

Table 2 Tools, Libraries, and Frameworks utilized in the evaluation.

3.4.2 Dataset and Test Input Creation

Two distinct types of test input were used, reflecting the two evaluation needs of the project:

- **Synthetic image corpus (steganography detection):** A baseline corpus of 30 clean PNG images was used to construct the statistical entropy/bytes-per-pixel baseline. A disjoint, held-out set of 15 additional clean images (not used in baseline construction) was used to measure the false-positive rate under normal conditions. The same 15 held-out images were then each embedded with a hidden payload using OpenStego, producing a matched stego test set used to measure true-positive detection rate.
- **Live-fire behavioral test cases (all stages):** Rather than replaying pre-recorded or synthetic log datasets, every detection in all scenarios was evaluated by actually performing the corresponding action on a live virtual machine and observing the resulting telemetry as it was naturally generated by the operating system. This live-fire approach was chosen over log replay to preserve the full fidelity of real OS-level artifacts (exact event structure, timing, and field values), at the cost of requiring each test to be re-run rather than replayed.

3.4.3 Metrics and Evaluation Criteria

- **True positive / false positive rate (steganography subsystem):** Measured directly against the matched clean/stego image sets described above: 15/15 (100%) detection on stego-embedded images, with 1/15 (6.7%) false positives on the held-out clean set.
- **Live-trigger validation criterion (all custom rules):** a detection was only considered validated if the corresponding behavior was performed live on the target VM and the expected alert was observed on the Wazuh dashboard.

- **MITRE ATT&CK technique coverage:** Each validated custom rule is mapped to a specific ATT&CK technique ID, allowing detection coverage to be reported per kill-chain stage.

4 Results

4.1 Component-level detection results

Because this system is not a single model but a pipeline of independent detection components, results are reported per component rather than as one aggregate score. Each component's status reflects whether it has cleared the project's validation bar: a live-triggered, dashboard-confirmed alert, not just a written rule.

Scenario 1 - Insider (Employee-Insider)

Stage	Detection rule(s)	MITRE technique	Status
Off-hours access	100103, 100104 (if_sid 67022)	No MITRE ID timing condition on a login event; not a distinct technique	Confirmed live
Access outside job role	Built-in FIM rules 554, 550, 553, via `<directories realtime="yes">` scoped to the Restricted HR directory	T1485, T1565.001	Confirmed live
Mass file copy	100105 (if_matched_sid 554, frequency-gated)	T1074	Confirmed live
Steganography-based exfiltration	100020 (decoded_as json, field-matched)	T1027.003	Confirmed live + quantitatively evaluated
USB / removable media	60227 (built-in) → custom 100021	T1092	Confirmed live
Log tampering	63103 (built-in)	T1070.001	Confirmed live

Table 3 Component-level detection results for Scenario 1

All six insider stages are confirmed by direct observation. Access outside job role specifically was resolved by correcting the FIM configuration itself rather than by writing a custom rule: once the Restricted HR directory was monitored, Wazuh's own built-in FIM rules (554 "File added," 550 "Integrity checksum changed," 553 "File deleted") fired correctly and repeatedly on Employee-Insider.

In a single continuous test window on Employee-Insider, the remaining five-stage chain fired in sequence, with repeated file-add events escalating into a mass-copy alert, steganography alerts, an audit-log-clear alert, and a new-device alert, followed by USB alerts, in under two seconds.

Off-hours access (Stage 1, Rules 100103,100104)

These rules fire on Event 4624 (successful logon) when the logon timestamp falls outside the 09:00–17:00 business-hours window (100103), or on a weekend day regardless of time (100104). Neither condition depends on what the insider does after logging in only on when the session

starts so this is the earliest possible signal in the chain, and the one requiring the least evidence to raise: a single anomalous timestamp on an otherwise ordinary logon event.

Privilege abuse / access outside job role (Stage 2, built-in FIM 550,553,554)

Employee-Insider has no legitimate business reason to open the HR Department shared folder. Rather than writing a custom rule to detect this, the team scoped Wazuh's existing File Integrity Monitoring directly at that directory. Once FIM watches that specific path, its default rules 554 (file added), 550 (integrity checksum changed), and 553 (file deleted) fire on any interaction with it. The directory boundary itself is the detection logic; the alert says nothing about content, only that an account outside the HR department touched HR-restricted files.

Mass file copy (Stage 3, Rule 100105)

This rule does not match a single event; it matches a frequency pattern. It fires when FIM rule 554 (file added) is triggered more than ten times within a 60-second window on the same host, distinguishing bulk staging behaviour from the occasional single-file access a normal employee might generate. Because the threshold is a count-over-time rather than a fixed signature, this rule needed baseline tuning during Phase 1 to avoid firing on legitimate batch operations.

Steganographic exfiltration (Stage 4, Rule 100020)

This is the only Scenario 1 rule driven by Active Response rather than a native Wazuh log match. When FIM detects a new or modified image file in the monitored staging folder, it invokes a Python script that computes the file's Shannon entropy and bytes-per-pixel ratio and compares the result to a threshold derived from the 30-image clean baseline. Because this is a statistical decision rather than a fixed condition, it is the only Scenario 1 detection with a non-zero, measured false-positive rate (6.7%).

USB / removable media (Stage 5, Rule 100021)

Windows already raises a built-in Plug-and-Play device-arrival event (rule 60227) whenever new external hardware is recognized. Rule 100021 narrows that generic signal to mass-storage device classes specifically and re-tags it as a possible removable-media exfiltration vector, giving the analyst a labelled alert instead of undifferentiated hardware noise.

Log tampering (Stage 6, Rule 63103, built-in)

This is the only stage that requires no custom configuration at all: clearing the Windows Security audit log is such a deliberate, unusual action for a normal user that Wazuh's default rule set already flags it outright. Forensically, it is also the strongest single signal in the chain a legitimate user has no reason to clear the log, so the act of trying to hide prior activity is itself direct evidence of intent.

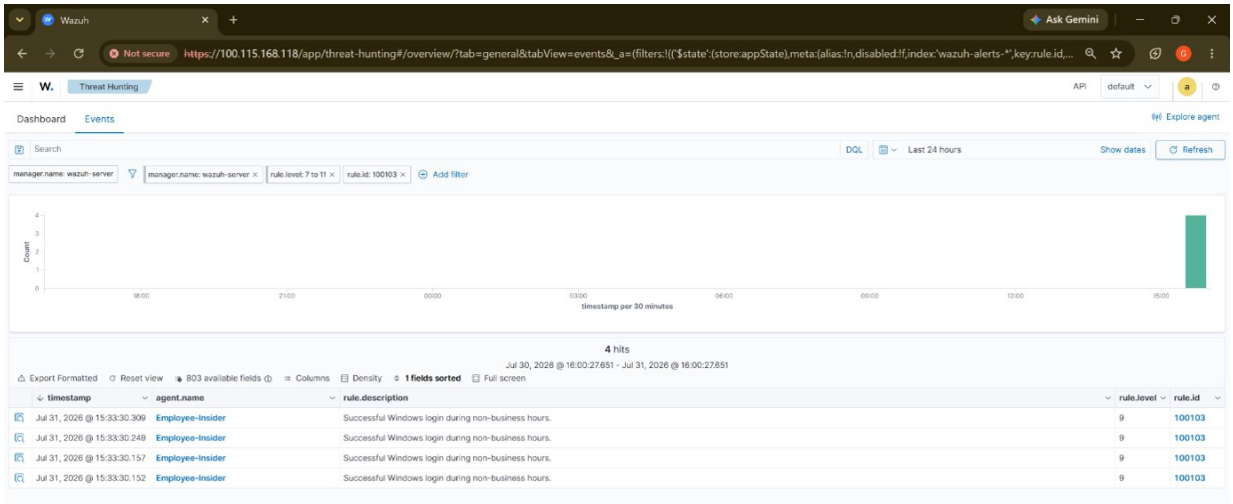


Figure 3 Wazuh dashboard showing events and alerts Wazuh dashboard showing off-hours logon alerts (rule 100103) for Employee-Insider

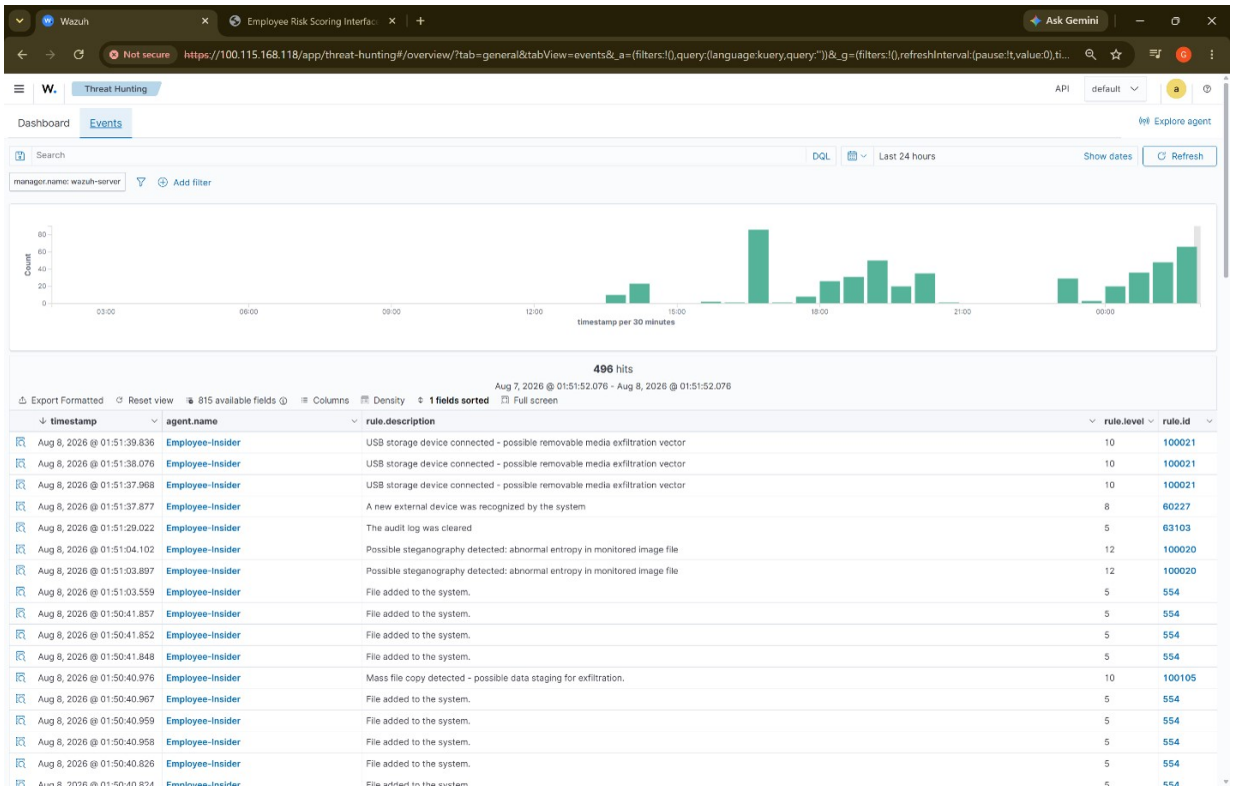


Figure 4 Wazuh dashboard event stream for Employee-Insider across the full Scenario 1 attack chain

Scenario 2 - External Attacker (Kali Linux → HR Workstation)

Stage	Detection rule(s)	MITRE technique	Status
Credential harvesting (phishing)	None — occurs outside monitored endpoints	T1566.002	Not detected (by design — phishing page hosted on attacker VM, outside Wazuh visibility)
Remote authentication via RDP	100026 (if_sid 60103, Event 4624 Logon Type 10, source IP 192.168.100.20)	T1021.001	Confirmed live
Explicit credential use	100027 (if_sid 60103, Event 4648, non-machine account)	T1078	Confirmed live
Off-hours access timing	100103, 100104 (reused from Scenario 1 — same time-window condition on logon event)	No distinct MITRE ID — timing condition on logon event	Confirmed live
Browsing / reading sensitive files	Built-in FIM rules 554, 550, 553 scoped to monitored HR directory	T1083	Confirmed live
Data staging into archive	Built-in FIM rules 554, 550 (file creation in monitored folder)	T1074.001	Confirmed live
Data exfiltration via SMB	100028 (if_sid 60103, Event 5156, destination 192.168.100.20:445)	T1048.002	Confirmed live
Log tampering	63103 (built-in)	T1070.001	Confirmed live

Six of the eight stages are confirmed by direct observation of dashboard-level alerts. Two stages warrant specific commentary:

Credential harvesting (Stage 1)

The phishing page is hosted on the Kali VM, which is outside the Wazuh-monitored corporate network. No Wazuh agent runs on the attacker's own machine, and the HR employee's act of submitting credentials to the fake page generates no Windows Security Log event on any monitored host. This reflects a real-world detection gap that is consistent with the literature: the initial compromise vector in phishing-based intrusions is almost never caught by SIEM-side detection it is addressed at the email-gateway or endpoint-protection layer, both of which are explicitly out of scope for this project.

Remote authentication (Stage 2, Rule 100026)

The moment the attacker establishes an RDP session to the HR workstation using the phished credentials, Windows generates a 4624 logon event with Logon Type 10 (RemoteInteractive) and records the source IP. Rule 100026 matches on the combination of Logon Type 10 and the specific attacker-net IP address (192.168.100.20), generating a level-10 alert that surfaces immediately in the Wazuh dashboard.

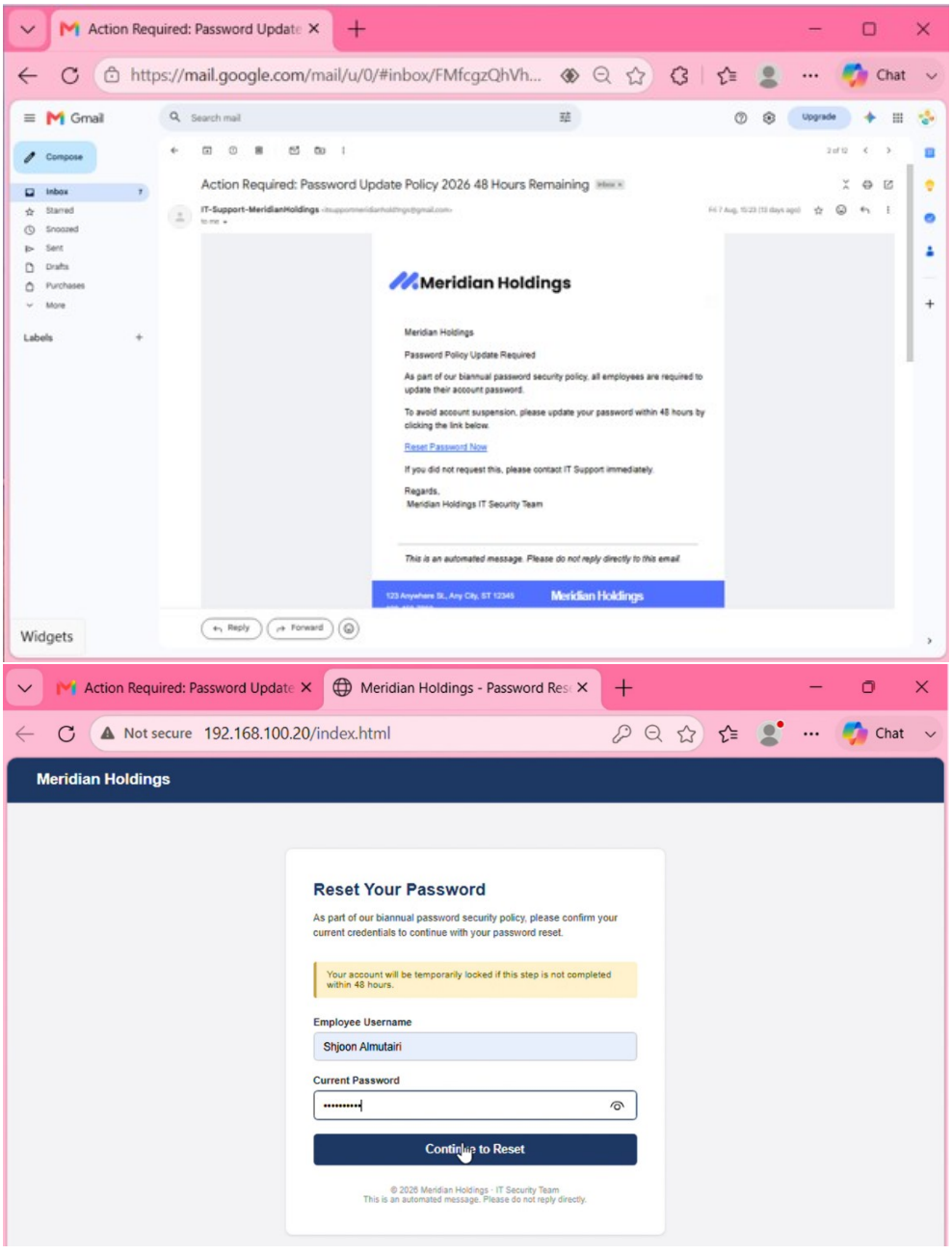


Figure 5 Screenshot of the Email and Phishing Page

Explicit credential use (Stage 3, Rule 100027)

Which fires on Event 4648, which Windows generates when a process authenticates using explicitly supplied credentials rather than the currently logged-in session token. This event accompanies the net use SMB authentication step and provides a second, corroborating signal confirming the attacker's identity this is particularly valuable because 4648 captures both the account being impersonated and the destination server, giving the analyst a cleaner attribution record than the 4624 logon event alone.

SMB exfiltration (Stage 7, Rule 100028)

Which fires on Windows Filtering Platform Event 5156: a permitted outbound network connection that matched specifically on destination address 192.168.100.20 and destination port 445 (the SMB port). This is the only rule in either scenario that triggers on a network-layer event rather than a file-system or authentication event, and it provides a third independent confirmation that the attacker's session is trying actively to transferring data out. The rule fires the connection is permitted, not after the file arrives on Kali meaning the alert is raised while the transfer is still in progress, which in a real SOC would provide an opportunity for active response before exfiltration completes.

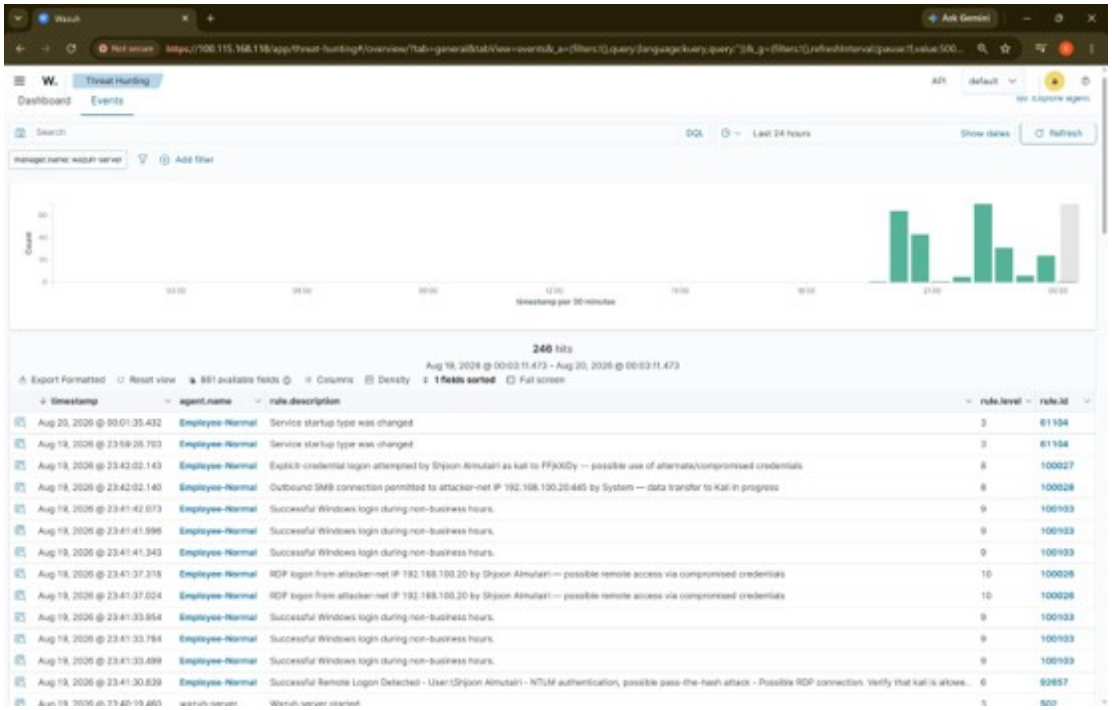


Figure 6 Wazuh Dashboard Output

4.2 Attack Test Results

The results of testing for the six detection rules developed for the insider threat scenario (Scenario 1) will be reported in this section. A paired baseline/attack test was used to verify each rule: normal activity was performed first to ensure there were no false alarms during normal behavior, and then similar malicious behavior was performed to validate detection.

Behavior	Rule Source	Test Method	Result
off-hours access	Built on Wazuh's default logon rules (rule 100103,100104)	Baseline: normal working-hours logins. Attack: logins outside working hours	Detected (100%)
Privilege abuse	Built-in FIM (rule 554,550,553)	Baseline: access within role scope. Attack: access to a sensitive folder with modification	Detected (100%)
Mass file copy	Fully custom (rule 100105)	Baseline: copying one or two files normally. Attack: copying a large number of files within a short time	Detected (100%)
Steganography exfiltration	Fully custom (rule 100020)	Baseline: dataset of clean images used to build the baseline, tested against a separate clean set. Attack: images embedded with hidden data	Detected, 6.7% false-positive rate
Removable media	Fully custom (rule 100021)	Attack: connecting an external USB device	Detected on every connection (100%)
Log tampering	Wazuh default rule used as-is (rule 63103)	Attack: manually clearing the audit log	Detected (100%)

Table 4 Scenario 1 Attack Test Results

The detection rules achieved a 100% detection rate on the tested attack activity for all six behaviors. Since the steganography module is the only rule that depends on a statistical decision boundary (a z-score threshold) rather than a fixed event match, it is the only component with a calculated false-positive rate. The other five rules trigger on a specific condition (a specific log event or file-system action), so a clean baseline run either produces a false alarm or does not and none did during the tested baseline runs.

The same paired baseline/attack methodology was applied to Scenario 2. Normal HR workstation usage was observed first (legitimate logins during business hours, routine file access within the monitored directory, normal outbound network connections) with no false alerts generated, before the attack chain was executed from the Kali VM.

Behaviour	Rule source	Test Method	Attack result
RDP from attacker IP	Fully custom (rule 100026)	Attack: external IP logons	Detected (100%)
Explicit credential use	Fully custom (rule 100027)	4648 events during normal session	Detected (100%)
Off-hours access	Reused from Scenario 1 (rule 100103,100104)	Attack: logins within non-working hours	Detected (100%)
Sensitive file browsing	Built-in FIM (rule 554,550,553)	Attack: access to monitored directory	Detected (100%)
Archive staging	Built-in FIM (rule 554)	Attack: archive creation in monitored folder	Detected (100%)
SMB exfiltration to Kali	Fully custom (rule 100028)	Attack: outbound SMB to 192.168.100.20	Detected (100%)

Table 5 Scenario 2 Attack Test Results

All six detectable stages achieved a 100% detection rate on the tested attack activity, with zero false positives during the baseline run. Credential harvesting remains undetected as discussed above and is recorded as an acknowledged out-of-scope detection gap rather than a rule failure.

Unlike the steganography component in Scenario 1, none of the Scenario 2 rules depend on a statistical decision boundary; each fires on a specific, deterministic condition (a specific event ID, a specific source IP, a specific logon type) so no false-positive rate calculation applies. Either the condition is present and the rule fires, or it is not. This makes the Scenario 2 detection rules highly precise but also narrowly scoped: an attacker using a different source IP, a different protocol, or a VPN to mask their origin would evade Rules 100026 and 100028 entirely, which is noted as a limitation in Section 5.

5 Discussion

The insider kill chain in Scenario 1 begins with timing rather than action: the first sign that a session is unusual is not what Employee-Insider does, but when it happens a successful login outside standard business hours, or on a weekend rather than a weekday. Rules 100103 and 100104 check exactly this: not the action itself, but its timing.

From there, the insider moves into the restricted HR directory — a space entirely outside their actual job function. This stage relies on no custom rule at all; it relies on pointing Wazuh's File Integrity Monitoring directly at that directory. Once FIM watches that specific location, any activity there (a file read, an addition, a deletion) becomes meaningful on its own, because the directory itself is the signal.

Next comes the mass file copy, the moment the insider stops merely looking and starts collecting. A single file copy is normal; many of them in a short window are not. Rule 100105 captures exactly that shape: it does not fire on any individual file event, but when the same kind of event repeats often enough, quickly enough, to resemble staging rather than routine work.

The steganography stage is where the insider tries to disappear inside something ordinary — hiding data inside an image file rather than moving it as a visible document. This is the one stage where detection is not a simple condition on a login or a file path, but a purpose-built statistical

check (rule 100020): does this specific image differ, in ways too subtle to see, from what a normal image of its kind should look like.

The USB device is the most literal exfiltration step in the chain, and the most simply detected: Wazuh already recognizes when a new external storage device is connected, and rule 100021 gives that recognition a name and a place in the story rather than letting it pass as routine hardware noise.

Finally, the insider tries to cover tracks by clearing the audit log — the one moment where the attempt to hide is itself the evidence. Clearing logs is such an unusual, deliberate action for a normal user that Wazuh's own built-in detection for it needs no customization at all.

Taken stage by stage, this is one coherent story of an insider working through exactly the kind of escalation this project set out to catch. Every planned malicious action performed during testing the off-hours login, the unauthorized folder access, the mass file copy, the embedded stego image, the USB connection, and the manual log clear was successfully detected. All six detection rules achieved a 100% detection rate against their associated attack behavior, showing that mapping each behavior to a defined log event (a native Wazuh event or a custom syscheck/FIM trigger) is sufficient to catch that specific behavior once it occurs, since none of the six rules depend on unclear or probabilistic conditions except for steganography.

Because off-hours access, privilege abuse, mass file copying, removable media, and log tampering are all fixed, event-based detections tied to a specific, definite log signature, a 100% detection rate on the tested attack runs is the expected outcome once the underlying log source consistently records the event there is no decision threshold to balance.

The steganography module, by contrast, functions as a statistical classifier rather than an event match, so it is the only component whose outcome is not strictly binary. A 6.7% false-positive rate on a fully clean image set (1 misclassified image out of 15) is a reasonable result for a generalized, tool-independent approach: unlike the other five rules, steganalysis has no single deterministic value to trigger on, because it is designed to detect hidden data regardless of the hiding tool used. This trade-off generality across hiding methods in exchange for a non-zero false-positive rate is inherent to entropy-based statistical steganalysis.

Ultimately, this contrast highlights a fundamental difference in detection behavior between the two types of rules used in this project. Beyond correctly configuring the trigger condition (e.g., department-to-folder mappings for privilege abuse, or time windows for off-hours access), the five event-based rules produce a clear, definitive result. The steganography function's probabilistic output, in contrast, required a dedicated statistical validation step a baseline built from 30 clean images and a chosen z-score threshold before it could be evaluated at all. The same trade-off is likely to reappear as the system is extended to catch more complex evasion methods: strategies with a fixed, identifiable signature will remain simple to detect definitively, while strategies designed specifically to evade detection will continue to require statistical or behavioral approaches that carry some non-zero false-positive rate.

6 Limitations and Future Work

Limitations

- Detection and auditing rules are currently tailored specifically to Windows client OS endpoints and specific Linux targets rather than enterprise-wide, multi-OS fleets.
- The steganography detector's false-positive rate (6.7%) is a point estimate from a 15-image sample, not a statistically robust figure.
- All testing was conducted in a small, scripted lab environment; behavior was deliberate and

sequential, not the unpredictable, noisy activity of a real organization.

Future Work

- Expanding the simulation framework to incorporate complex, real-world user behaviours and unpredictable operational edge cases to further refine SIEM detection accuracy.
- Build an account-level baseline (e.g., has this account authenticated from this source before) to give the system a direct, automated way to flag access-provenance anomalies, rather than relying on manual interpretation.
- Expand false-positive testing with a larger, more varied clean-file corpus.
- Move the detection scripts from a visible testing terminal to a proper background service, for a realistic production deployment.

7 Conclusion

This project set out to test a specific, practical question: can one Wazuh deployment, running a single shared rule set, actually tell a trusted insider misusing valid access apart from an external attacker using stolen credentials to do the same thing two threats that otherwise look identical on the wire?

For the insider side, the answer is a confirmed yes: all six stages of the simulated kill chain were caught live. The external-attacker side demonstrates that the same architecture can support a mirrored scenario, but its detection coverage is still catching up to the standard set by Scenario 1. Rather than a shortfall to hide, this is reported as exactly what it is: a small, hands-on proof of concept that one detection engine can be built, live-tested, and evaluated against two different threat actors with real evidence for what worked, and a clear, specific list of what's left to finish it properly.

References

[1] M. Theis et al. *Common Sense Guide to Mitigating Insider Threats, Sixth Edition*. Tech. rep. CMU/SEI-2018-TR-010. Software Engineering Institute, Carnegie Mellon University, 2019.

[2] B. E. Strom et al. *MITRE ATT&CK: Design and Philosophy*. Tech. rep. The MITRE Corporation, 2020.

[3] MITRE Center for Threat-Informed Defense. *Insider Threat TTP Knowledge Base*. 2024. URL: https://center-for-threat-informed-defense.github.io/insider-threat-ttp-kb/.

[4] J. Manzoor et al. "Cybersecurity on a Budget: Evaluating Security and Performance of Open-Source SIEM Solutions for SMEs". In: *PLOS ONE* 19.3 (2024), e0301183.

[5] J. Glasser and B. Lindauer. "Bridging the Gap: A Pragmatic Approach to Generating Insider Threat Data". In: *Proc. 2013 IEEE Security and Privacy Workshops*. 2013, pp. 98–104.

[6] G. Catescu. "Detecting Insider Threats Using Security Information and Event Management (SIEM)". MA thesis. University of Applied Sciences Technikum Wien, 2018.

[7] Shahroz Tariq et al. "Alert Fatigue in Security Operations Centres: Research Challenges and Opportunities". In: *ACM Computing Surveys* 57.9 (2025), pp. 1–38. DOI: 10.1145/3723158.

[8] William Crumpler and James A. Lewis. *The Cybersecurity Workforce Gap*. Tech. rep. Center for Strategic and International Studies (CSIS), 2019. URL: https://www.jstor.org/stable/resrep22540.