



SOC INCIDENT REPORT

SOAR-Lite-Splunk-SSH-Bruteforce

1. Executive Summary (Management View)

During routine security monitoring, multiple failed SSH authentication attempts were detected on a Linux system. The activity originated from a single source IP and exceeded normal baseline behavior.

The event was identified using Splunk SIEM, enriched using threat intelligence (AbuseIPDB), and analyzed via a Python-based SOAR-Lite automation pipeline.

Based on enrichment results and contextual analysis, the activity was classified as **low-severity brute-force behavior in a controlled lab environment**, requiring no immediate containment. This incident demonstrates effective detection, validation, and analyst decision-making while avoiding unnecessary escalation.

2. Why This Alert Matters (Analyst Thinking)

SSH is a **high-value access vector** in Linux environments.

Repeated authentication failures can indicate:

- Credential stuffing
- Password brute-force attempts
- Unauthorized access attempts

However, **not every failed login is malicious**.

As an analyst, I must determine **intent, risk, and impact** before escalation.

3. Detection Source & Data Collection

Log Source

- System: Ubuntu Linux
- Log file: `/var/log/auth.log`
- Service monitored: `sshd`

SIEM Tool

- Splunk Enterprise (Free)

Detection Logic

- Failed SSH authentication events (`Failed password`)
- Aggregation of failures by source IP

Evidence:

- Log ingestion configuration
- Failed SSH log entries
- Event-level details
- Aggregated statistics by attacker IP

Add Data

Select SourceSet Source TypeInput SettingsReviewDone

< Back

Next >

Files & Directories

Upload a file, index a local file, or monitor an entire directory.

HTTP Event Collector

Configure tokens that clients can use to send data over HTTP or HTTPS.

TCP / UDP

Configure the Splunk platform to listen on a network port.

Scripts

Get data from any API, service, or database with a script.

checkapp

Systemd Journal Input for Splunk

This is the input that gets data from journald (systemd's logging component) into Splunk.

Logd Input for the Splunk platform

This input collects data from logd on macOS and sends it to the Splunk platform.

Splunk Secure Gateway

Initializes the Splunk Secure Gateway application to talk to mobile clients over websockets

Splunk Secure Gateway Mobile Alerts TTL

Configure this instance to monitor files and directories for data. To monitor all objects in a directory, select the directory. The Splunk platform monitors and assigns a single source type to all objects within the directory. This might cause problems if there are different object types or data sources in the directory. To assign multiple source types to objects in the same directory, configure individual data inputs for those objects. [Learn More](#)

File or Directory ?

Browse

On Windows: c:\apache\apache.error.log or \\hostname\apache\apache.error.log. On Unix: /var/log or /mnt/www01/var/log.

Continuously Monitor

Index Once

Include list ?

Exclude list ?

FAQ

- > What kinds of files can the Splunk platform index?
- > I can't access the file that I want to index. Why?
- > How do I get remote data onto my Splunk platform instance?
- > Can I monitor changes to files in addition to their content?

Timeline format
Zoom Out
Zoom In
Zoom to Selection
DeSelect

Format
Show: 20 Per Page
View: List

Prev
1
2
Next

Hide Fields

All Fields

SELECTED FIELDS

a host 1

a source 1

a sourcetype 1

INTERESTING FIELDS

date_hour 1

date_mday 1

date_minute 5

date_month 1

date_second 26

a date_wday 1

date_year 1

date_zone 1

a index 1

linecount 1

pid 10

a process 1

a punct 1

a splunk_server 1

timeendpos 1

timestartpos 1

+ Extract New Fields

i	Time	Event
>	2/4/26 6:46:40.586 PM	2026-02-04T18:46:40.586315+00:00 Ubuntu sshd[71077]: Failed password for invalid user test from 192.168.56.104 port 42854 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:34.556 PM	2026-02-04T18:46:34.556704+00:00 Ubuntu sshd[71026]: Failed password for invalid user test from 192.168.56.104 port 51676 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:30.302 PM	2026-02-04T18:46:30.302005+00:00 Ubuntu sshd[71026]: Failed password for invalid user test from 192.168.56.104 port 51676 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:28.953 PM	2026-02-04T18:46:28.953922+00:00 Ubuntu sshd[71026]: Failed password for invalid user test from 192.168.56.104 port 51676 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:20.181 PM	2026-02-04T18:46:20.181848+00:00 Ubuntu sshd[71009]: Failed password for invalid user test from 192.168.56.104 port 51328 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:14.976 PM	2026-02-04T18:46:14.976702+00:00 Ubuntu sshd[71009]: Failed password for invalid user test from 192.168.56.104 port 51328 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:09.380 PM	2026-02-04T18:46:09.380974+00:00 Ubuntu sshd[71009]: Failed password for invalid user test from 192.168.56.104 port 51328 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:04.708 PM	2026-02-04T18:46:04.708556+00:00 Ubuntu sshd[70958]: Failed password for invalid user test from 192.168.56.104 port 36698 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:46:00.313 PM	2026-02-04T18:46:00.313121+00:00 Ubuntu sshd[70958]: Failed password for invalid user test from 192.168.56.104 port 36698 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:45:48.954 PM	2026-02-04T18:45:48.954356+00:00 Ubuntu sshd[70958]: Failed password for invalid user test from 192.168.56.104 port 36698 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure
>	2/4/26 6:45:41.475 PM	2026-02-04T18:45:41.475399+00:00 Ubuntu sshd[70916]: Failed password for invalid user test from 192.168.56.104 port 56330 ssh2 host = Default source = /var/log/auth.log sourcetype = linux_secure

6:46:30.302 PM

2/4/26

6:46:28.953 PM

Event Actions

Type	Field	Value	Actions
Selected	host	Default	
	source	/var/log/auth.log	
	sourcetype	linux_secure	
Event	pid	71026	
	process	sshd	
Time	_time	2026-02-04T18:46:28.953+00:00	
Default	index	main	
	linecount	1	
	punct	--::+:_[]:_____	
	splunk_server	Ubuntu	

4. Noise vs Risk Analysis

Observed Behavior

- Multiple failed login attempts
- Single source IP
- Short time window

Potential Noise Indicators

- No successful authentication
- No lateral movement
- No privilege escalation
- IP not previously flagged

Potential Risk Indicators

- Repeated authentication attempts
- Use of SSH (remote access service)

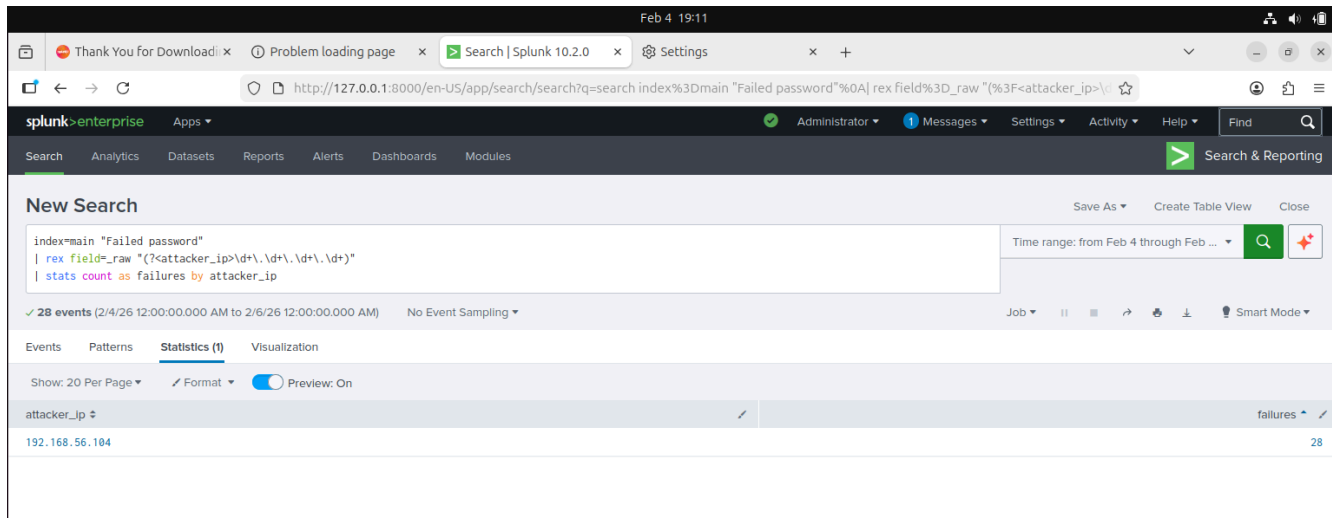
5. Event Aggregation & Alert Review in Splunk

Splunk was used to:

- Identify failed SSH attempts
- Group events by attacker IP
- Quantify attack volume

Detection Query (Conceptual)

- Failed authentication events
- Count by source IP
- Threshold-based review



6. Incident Context Enrichment (Why SOAR Is Required)

SIEM answers:

“What happened?”

SOAR + Threat Intel answers:

“How dangerous is it?”

Enrichment Objective

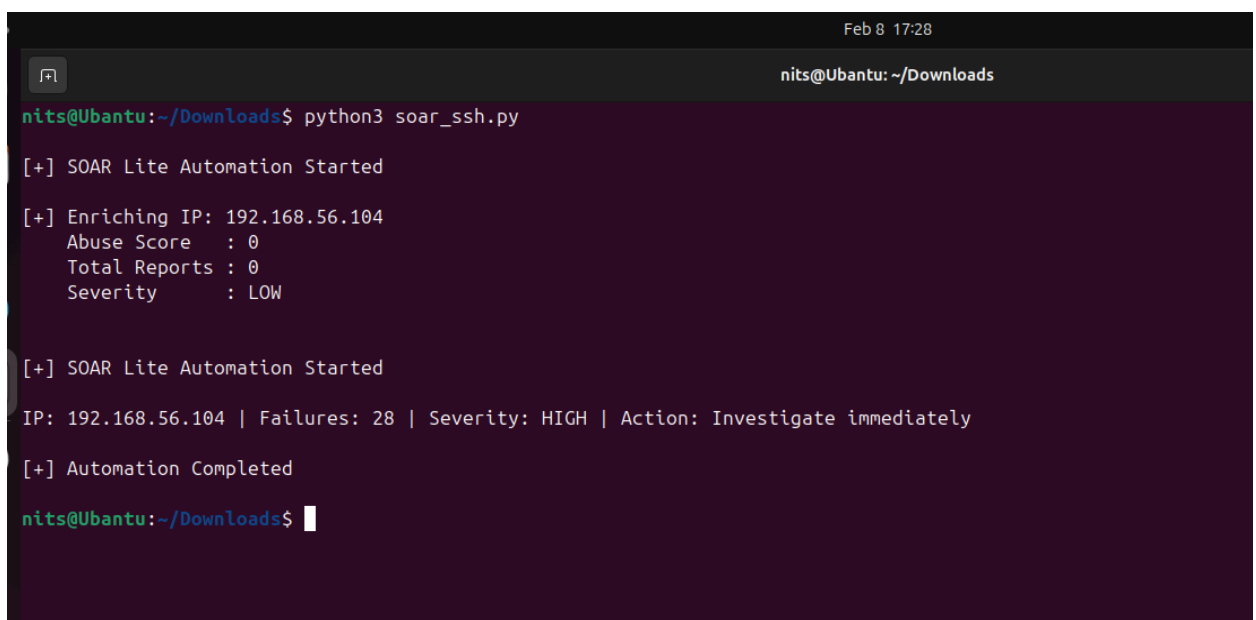
- Validate whether the attacking IP is **known malicious**
- Reduce false positives
- Support analyst decision-making

Threat Intelligence Source

- AbuseIPDB

Enrichment Data Retrieved

- Abuse confidence score
- Number of reports
- Reputation status



```
Feb 8 17:28
nits@Ubuntu: ~/Downloads

nits@Ubuntu:~/Downloads$ python3 soar_ssh.py

[+] SOAR Lite Automation Started

[+] Enriching IP: 192.168.56.104
    Abuse Score   : 0
    Total Reports : 0
    Severity      : LOW

[+] SOAR Lite Automation Started

IP: 192.168.56.104 | Failures: 28 | Severity: HIGH | Action: Investigate immediately

[+] Automation Completed

nits@Ubuntu:~/Downloads$
```

7. SOAR-Lite Automation (Analyst Efficiency)

A Python automation script was used to:

- Ingest SIEM-exported CSV alerts
- Extract attacker IPs
- Query AbuseIPDB API
- Assign severity based on reputation

- Produce analyst-readable output

Why Automation Matters in SOC

- Reduces manual lookups
- Ensures consistency
- Saves analyst time
- Prevents alert fatigue

8. Severity Classification & Decision Logic

Enrichment Results

- Abuse Confidence Score: Low
- Total Reports: None / Minimal
- IP Reputation: Clean

Final Severity

LOW

Justification

- No malicious reputation
- Lab-generated activity
- No signs of compromise

As analyst Decision:

No escalation required. Monitor for recurrence.

9. Incident Handling & Response Actions

Actions Taken

- Alert investigated
- IP behavior validated
- Severity assigned
- Incident documented

Actions Not Taken (Justified)

- No IP blocking
- No account lockout
- No escalation to IR team

Reason:

Response must be proportional to risk.

10. Communication & Escalation Evidence

Internal SOC Communication (Simulated)

Status Update (Tier-1 → Tier-2):

“Repeated SSH authentication failures detected from a single IP. Threat intelligence enrichment shows no malicious reputation. Severity classified as LOW. No further action required unless behavior persists.”

11. Lessons Learned (As Analyst)

- Detection alone is insufficient

- Context determines severity
- Threat intelligence reduces noise
- Automation enhances analyst effectiveness
- Not every alert is an incident

12. Limitations & Future Improvements

Current Limitations

- Single-IP lab environment
- No real external attackers

Future Enhancements

- VirusTotal enrichment
 - GeolIP analysis
 - Automated response actions
 - Case management integration
 - Multi-source correlation
-

13. Conclusion

This project demonstrates an end-to-end **SOC detection and response workflow**, integrating SIEM monitoring, automation, threat intelligence enrichment, and analyst decision-making.

It reflects how modern SOC teams balance alert volume, accuracy, and operational efficiency.

