Table of Contents

List of Figures

List of Tables

1.Introduction

In today's evolving cybersecurity landscape, organisations must proactively monitor and analyse system logs to detect and respond to security threats effectively. A Security Operations Centre (SOC) plays a crucial role in safeguarding an organisation's digital infrastructure by continuously assessing logs for anomalies, potential intrusions, and suspicious activities.

This study aims to deploy Splunk for comprehensive log analysis across multiple system environments, focusing on the detection of security incidents and operational anomalies. The primary objectives include:

- Centralising log data from various sources to enhance visibility.
- Identifying potential security threats, including brute-force attacks, unauthorised access attempts, and unusual user behaviours.
- Utilising Splunk's visualisation capabilities to generate insightful dashboards for real-time monitoring.
- Providing mitigation strategies based on findings to strengthen organisational security.

The anticipated outcome of this study is the successful implementation of Splunk as a log analysis and security monitoring tool, enabling security teams to make data-driven decisions and improve incident response times. Additionally, the study aims to highlight areas for future enhancements, including automation and integration with advanced security intelligence platforms.

2. Technology Comparison: Alternative Tools for Log Analysis

Feature	**Splunk**	**ELK Stack**	**Graylog**	**QRadar**	**LogRhythm**
Ease of Deployment	Easy, fast setup	Complex, manual setup	Moderate, requires Graylog server	Enterprise-grade, requires expertise	Structured but complex
User Interface	Intuitive, customizable dashboards	Requires Kibana setup	Simple web UI	Prebuilt SOC templates	Compliance-focused UI
Security Features	AI-driven threat detection	Requires external tools	Basic security logs	Advanced SIEM, AI analytics	SIEM with UEBA tools
Alerting & Automation	Built-in SOAR, automated alerts	Basic alerting, external tools needed	Integrated alerts	Strong automation & correlation	Automated incident response
Scalability	High, but costly at large volumes	Great for unstructured data	Moderate, supports clustering	Large-scale enterprises	Scales well, optimized for security data
Best For	AI-driven SOCs, real-time security monitoring	Large-scale log aggregation	Mid-sized SOCs, simple log monitoring	Enterprise SIEM with AI	Compliance & security-focused SOCs
Cost	High (based on data volume)	Open-source (lower cost, more expertise needed)	Free & enterprise versions	Expensive but feature-rich	Premium pricing, AI-driven

Table 1: Splunk vs ELK vs Graylog vs QRadar vs LogRhythm

3. Results and Visualisations

To enhance security monitoring and threat detection, various dashboards were created in Splunk to visualise log data effectively. These visualisations allow security analysts to gain real-time insights into system activity, identify threats, and take prompt action to mitigate risks. Below are the key visualisations used:

1.Apache Web Server Log

1.1 Countries with the Most Requests

Command: index=* source="Apache.log" host="windows" sourcetype="apache"

| rex field=_raw "\[client (?<client_ip>\d+\.\d+\.\d+\.\d+)\]"

| iplocation client_ip

| geostats count by Country latfield=lat longfield=lon

To analyze web traffic and detect potential threats, this Splunk query extracts client IP addresses from Apache logs, maps them to geographical locations, and visualizes traffic using a choropleth map. The query filters relevant logs, extracts IPs using the rex command, and applies iplocation to determine their origin. The geostats function then plots request counts by location. This helps identify abnormal traffic patterns, such as botnet activity or unusual spikes, enabling security teams to detect and respond to potential threats effectively.

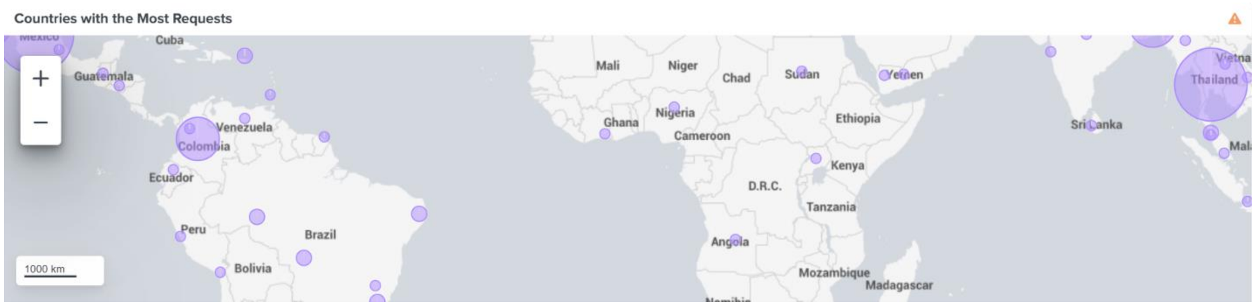


Figure 1: Apache – Countries with the Most Requests

1.2 Peak Traffic Hours Panel

Command : index=* source="Apache.log" host="windows" sourcetype="apache"

| bin _time span=1h

| stats count as "Total Requests" by _time

| sort _time

The command retrieves Apache logs from a Windows host, groups requests into hourly intervals, and calculates the total requests per hour. It then sorts the data chronologically to visualize peak traffic hours. This helps detect unusual spikes—legitimate traffic aligns with business hours, while high off-hour activity may indicate automated attacks or unauthorized access attempts.

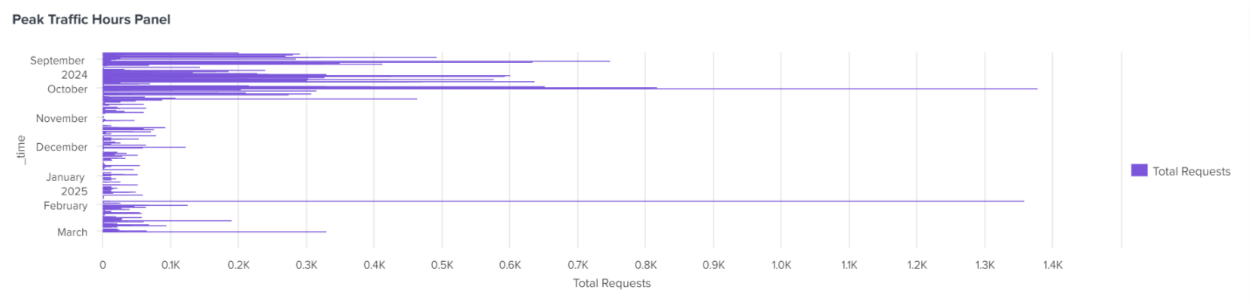


Figure 2: Apache – Peak Traffic Hours Panel

1.3 Requests Over Time

Command : index=* source="Apache.log"

| bin _time span=1h

| stats count as "Total Requests" by _time

The command retrieves all Apache log data, groups requests into hourly intervals, and calculates the total requests per hour. This time-series chart helps track request frequency, identifying abnormal spikes that may indicate DDoS attacks, brute-force attempts, or data scraping bots.

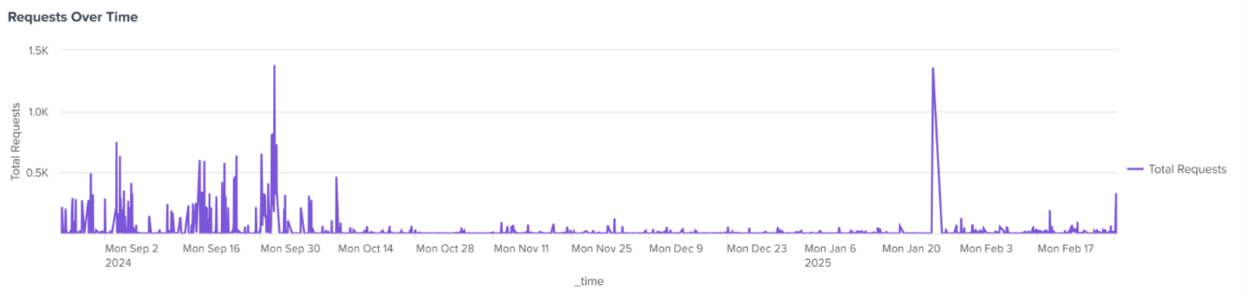


Figure 3: Apache – Requests Over Time

1.4 Top 10 Client IPs Making Requests

Command : index=* source="Apache.log" host="windows" sourcetype="apache"

| rex field=_raw "\[client (?<client_ip>\d+\.\d+\.\d+\.\d+)\]"

| stats count by client_ip

| sort - count

| head 10

The command extracts Apache logs, counts requests per IP, and lists the top 10 sources. A bar chart helps identify botnet activity if a few IPs dominate traffic.

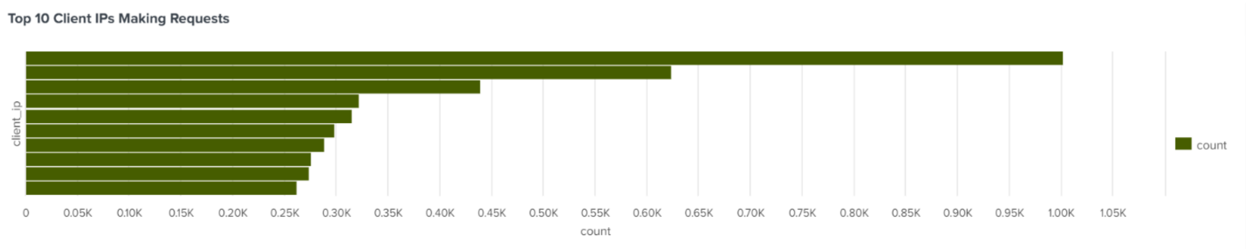


Figure 4: Apache – Top 10 Client IPs Making Requests

1.5 Unique Visitors Over Time

Command : index=* source=”Apache.log” host=”windows” sourcetype=”apache”

| rex field=_raw “\[client (?<client_ip>\d+\.\d+\.\d+\.\d+)\]”

| timechart span=1h dc(client_ip) as “Unique Visitors”

The command retrieves Apache logs, extracts client Ips, and tracks unique visitors per hour. Gradual growth indicates normal usage, while sudden spikes may signal credential stuffing attacks or bot activity.

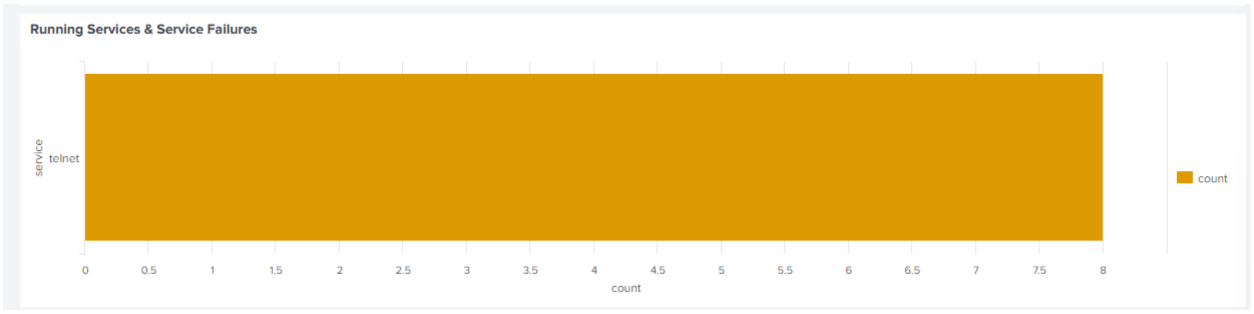


***Figure 5: Apache – Unique Visitors Over Time*2 This Mac Logs**

2. Mac OS log

2.1 System Wake Reasons

Command : index=* source="Mac.log" host="macos_logs" sourcetype="macos_system"

| search "Wake reason"

| rex field=_raw "Wake reason: (?<wake_reason>.*)"

| stats count by wake_reason

The command retrieves macOS logs, extracts wake reasons, and counts their frequency. A pie chart helps detect unusual wake patterns, indicating unauthorized access or malware activity.

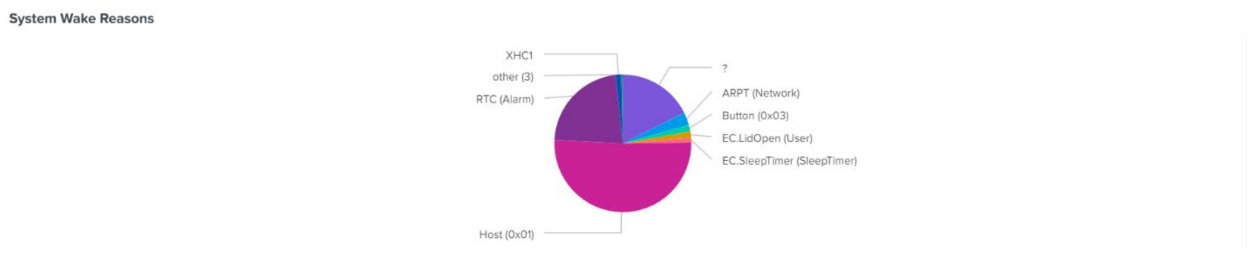


Figure 6: Mac – System Wake Reasons

2.2 Wi-Fi Connection & Disconnection Logs:

Command: index=* source="Mac.log" host="macos_logs" sourcetype="macos_system"

| search "AirPort: Link"

| rex field=_raw "AirPort: (?<wifi_event>.*)"

| stats count by wifi_event

The command retrieves macOS logs, extracts Wi-Fi connection and disconnection events, and counts occurrences. A bar chart helps track network instability, rogue access points, or blocked unauthorized connections.

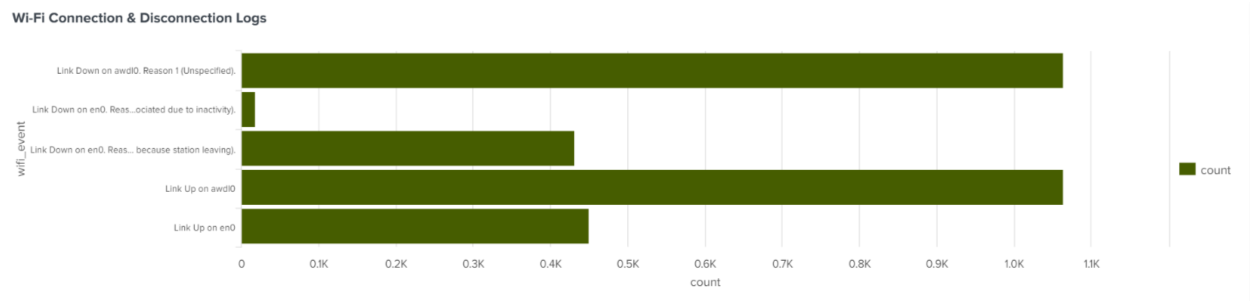


Figure 7: Mac – Wi-Fi Connection & Disconnection Logs

2.3 Wi-Fi Connection Failures Over Time

Command : index=* source=”Mac.log” host=”macos_logs” sourcetype=”macos_system”

| search “AirPort: Link Down”

| timechart span=1h count

The command retrieves macOS logs, tracks Wi-Fi disconnections, and visualizes failures per hour. A time-series chart helps detect network instability, recurring issues, or deauthentication attacks.

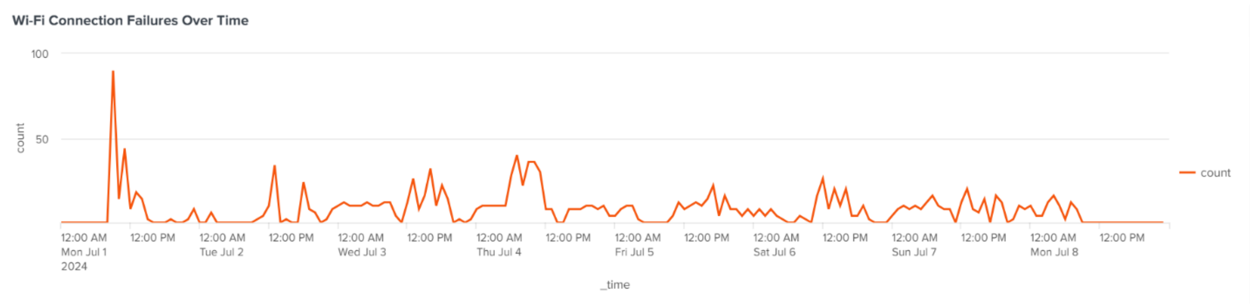


Figure 8: Mac – Wi-Fi Connection Failures Over Time

2.4 USB & Thunderbolt Device Trends

Command : index=* source=”Mac.log” host=”macos_logs” sourcetype=”macos_system”

| search “USBMSC Identifier” OR “Thunderbolt”

| rex field=_raw “(?i)(USBMSC Identifier|Thunderbolt) (?<device_name>.+)”

| timechart span=1h count by device_name

The command retrieves macOS logs, tracks USB and Thunderbolt connections, and visualizes activity per hour. A time-based chart helps detect unusual spikes, data exfiltration attempts, or unauthorized hardware access.

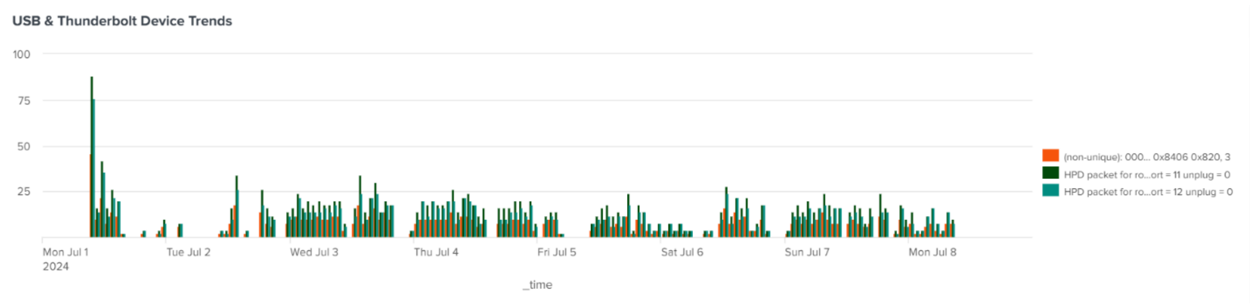


Figure 9: Mac – USB & Thunderbolt Device Trends

2.5 System Camera Activity

Command : index=* source=”Mac.log” host=”macos_logs” sourcetype=”macos_system”

| search “AppleCamIn::”

| timechart span=1h count

The command retrieves macOS logs, tracks webcam activations, and visualizes usage per hour. A time-series chart helps detect unauthorized access, privacy breaches, or malware activity, especially outside working hours.

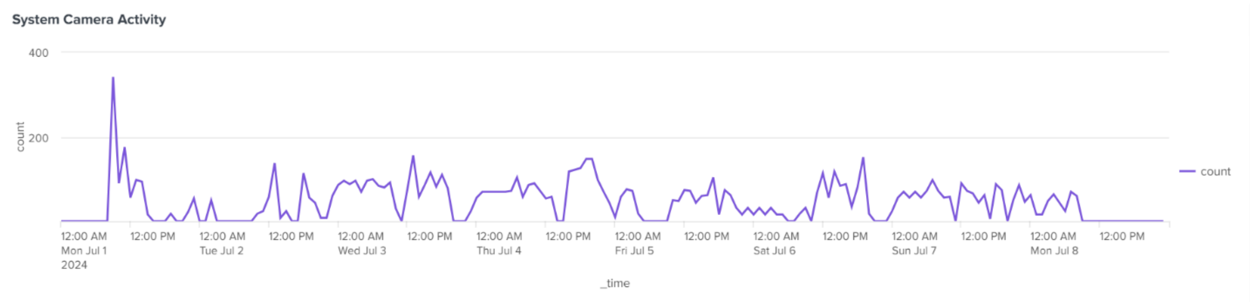


Figure 10: Mac – System Camera Activity

3 Linux System Log

3.1 Active Network Connections

Command: index=* source="Linux.log" host="linuxlog" sourcetype="linux"

| search "listening on IPv4 interface"

| rex field=_raw "interface (?<interface>\w+), (?<ip>\d+\.\d+\.\d+\.\d+)#\d+"

| stats count by ip

The command retrieves Linux logs, tracks active network interfaces, and maps IP occurrences. A choropleth map helps detect unauthorized access, traffic spikes, or malicious activity from unusual locations.

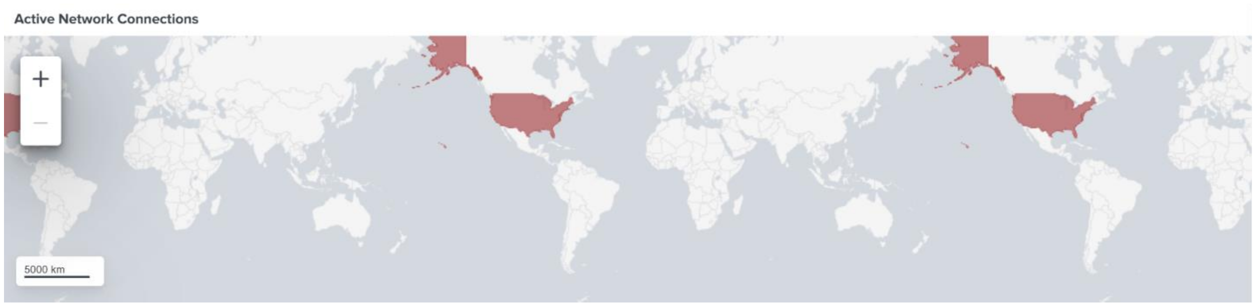


Figure 11: Linux – Active Network Connections

3.2 Running Services & Service Failures

Command: index=* source="Linux.log" host="linuxlog" sourcetype="linux"

| search "startup succeeded" OR "failed"

| rex field=_raw "(?<service>\w+): (startup succeeded|failed)"

| stats count by service

The command retrieves Linux logs, tracks service startups and failures, and visualizes occurrences in a bar chart. This helps SOC teams detect critical disruptions, misconfigurations, or potential system compromises.

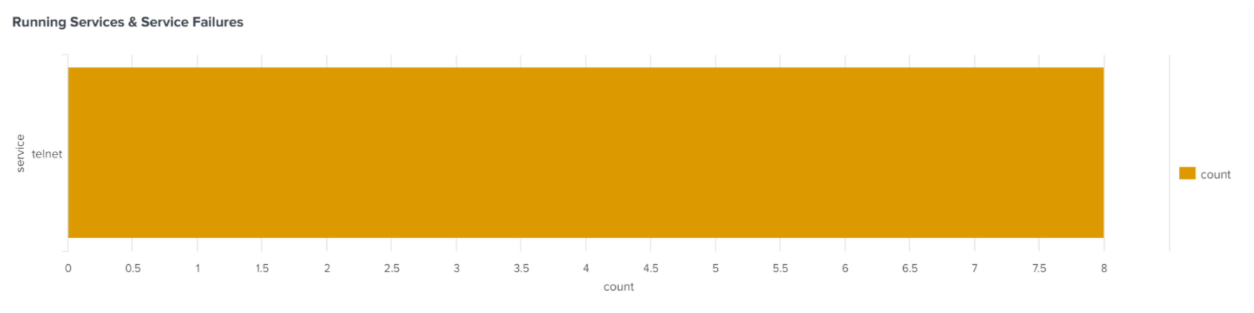


Figure 12: Linux – Running Services & Service Failures

3.3 System Reboots & Startups

Command: index=* source="Linux.log" host="linuxlog" sourcetype="linux"

| search "syslogd startup succeeded" OR "reboot"

| timechart span=1d count

The command retrieves Linux logs, tracks system startups and reboots, and visualizes occurrences daily. A time-series graph helps detect hardware failures, unauthorized restarts, or malware persistence attempts, ensuring system stability and security.

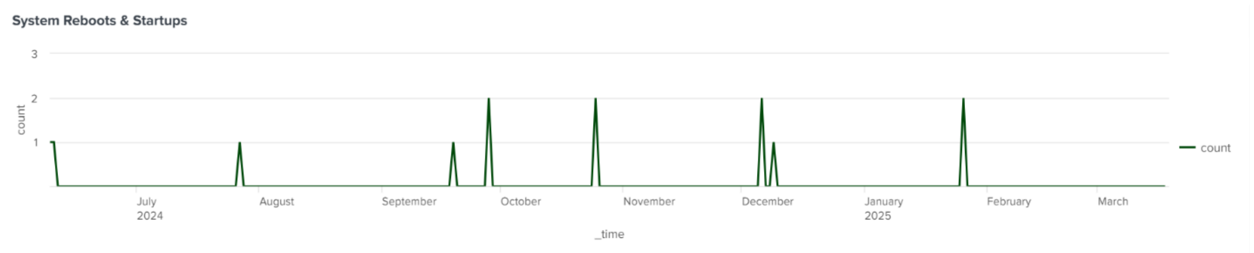


Figure 13: Linux – System Reboots & Startups

3.4 System Time Synchronisation Issues

Command: index=* source="Linux.log" host="linuxlog" sourcetype="linux"

| search "ntpd"

| timechart span=1h count

The command retrieves Linux logs, tracks NTP-related events, and visualizes occurrences per hour. A time-series graph helps detect time synchronization issues, misconfigurations, or cyber-attacks manipulating timestamps, ensuring accurate system logs and forensic integrity.

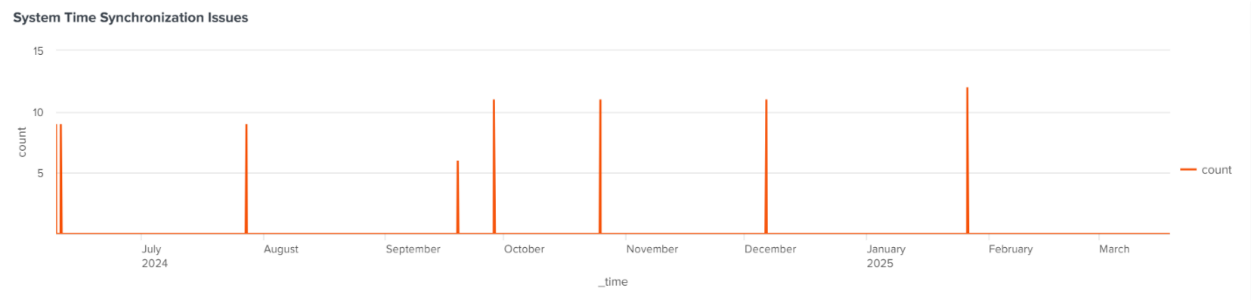


Figure 14: Linux – System Time Synchronisation Issues

3.5 USB & Hardware Connection Events

Command: index=* source="Linux.log" host="linuxlog" sourcetype="linux"

| search "USB" OR "device node"

| rex field=_raw "USB device: (?<device_name>.+)"

| timechart span=1h count by device_name

The command retrieves Linux logs, tracks USB and hardware connections, and visualizes occurrences per hour. A time-series graph helps detect unauthorized devices, data exfiltration attempts, or malicious firmware activity, ensuring security and compliance.

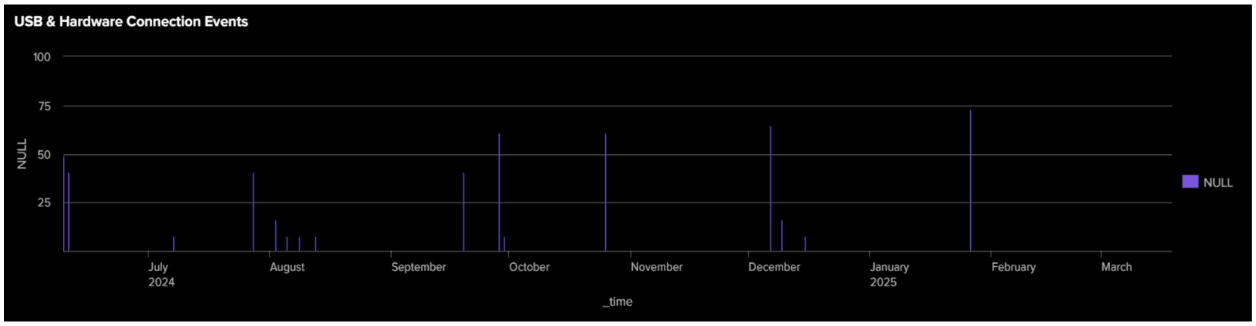


Figure 15: Linux – USB & Hardware Connection Events

4 Health App Log

4.1 Unusual Activity at Midnight:

Command: index=* source="HealthApp.log" host="app" sourcetype="health"

| rex field=_raw "setTodayTotalDetailSteps=.*##(?<total_steps>\d+)##"

| eval hour=strftime(_time, "%H")

| search hour >= "00" AND hour <= "06"

| stats count by hour

The command retrieves health app logs, extracts step counts, and filters activity between midnight and 6 AM. A time-based analysis helps detect automated processes, data modifications, or suspicious user behavior requiring further investigation.

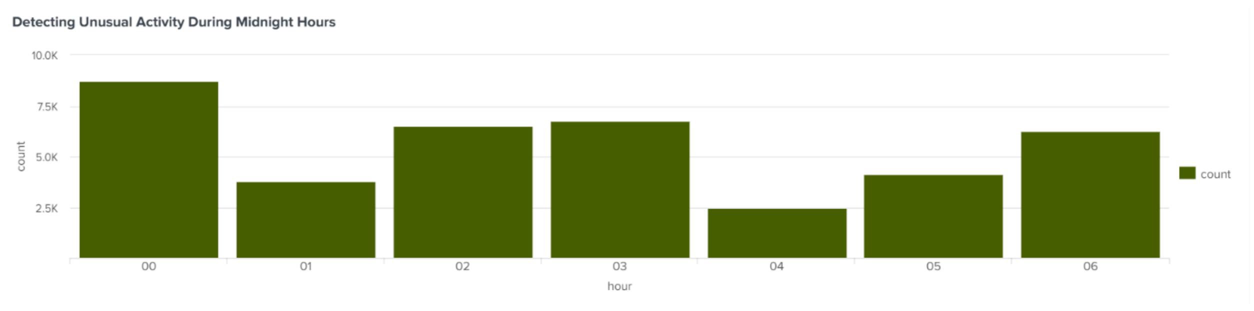


Figure 16: Health App – Unusual Activity at Midnight

4.2 Data Integrity Issues

Command: index=* source="HealthApp.log" host="app" sourcetype="health"

| rex field=_raw "setTodayTotalDetailSteps=.*##(?<total_steps>\d+)##"

| stats count by total_steps

The command retrieves health app logs, extracts step counts, and analyzes their frequency. This helps detect data inconsistencies, tampering, logging errors, or unauthorized modifications, ensuring data integrity and accuracy.

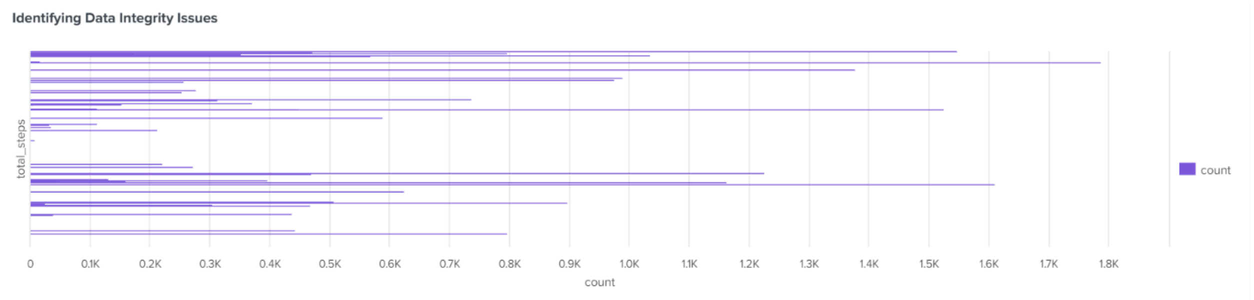


Figure 17: Health App – Data Integrity Issues

4.3 User Activity by Weekday

Command: index=* source="HealthApp.log" host="app" sourcetype="health"

| rex field=_raw "setTodayTotalDetailSteps=.*##(?<total_steps>\d+)##"

| eval weekday=strftime(_time, "%A")

| stats avg(total_steps) by weekday

The command retrieves health app logs, extracts step counts, and tracks average activity per weekday. This helps identify user engagement trends, detecting sudden drops, spikes, or irregular usage that may indicate technical issues or health concerns.

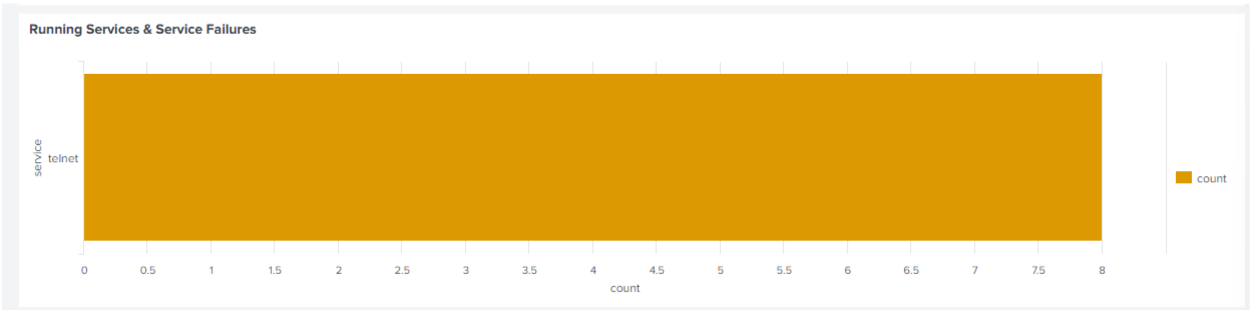


Figure 18: Health App – User Activity by Weekday

4.4 Steps vs. Screen Activity

Command: index=* source="HealthApp.log" host="app" sourcetype="health"

| rex field=_raw "setTodayTotalDetailSteps=.*##(?<total_steps>\d+)##"

| eval screen_status=if(like(_raw, "%android.intent.action.SCREEN_ON%"), "Screen On", "Screen Off")

| stats avg(total_steps) by screen_status

The command retrieves health app logs, extracts step counts, and categorizes activity by screen on/off status. This helps detect prolonged screen time without movement, indicating device misuse, inactivity, or automated background activity.

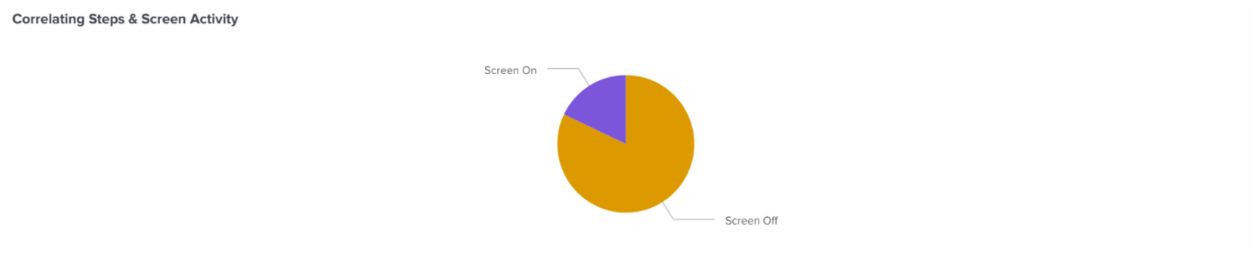


Figure 19: Health App – Steps vs. Screen Activity

4.5 Frequent Screen Locking

Command: index=* source="HealthApp.log" host="app" sourcetype="health"

| search "onReceive action: android.intent.action.SCREEN_OFF"

| bucket _time span=10m

| stats count by _time

The command retrieves health app logs, tracks screen lock events, and groups them into 10-minute intervals. This helps detect frequent screen locks, indicating security-conscious behavior or potential device compromise requiring repeated authentication attempts.

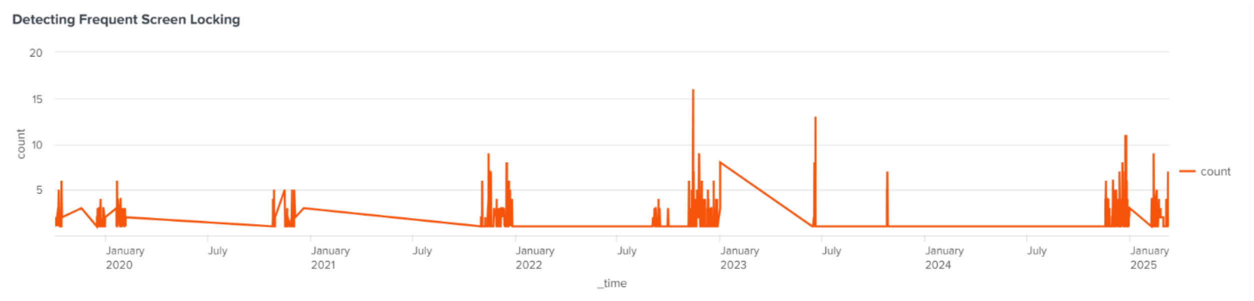


Figure 20: Health App – Frequent Screen Locking

5 OpenSSH server log

5.1 SSH Brute-Force Attacks

Command: index=* source="OpenSSH server.log" host="openssh" sourcetype="openssh server"

| search "Failed password"

| rex field=_raw "Failed password for (?<username>\S+) from (?<ip>\d+\.\d+\.\d+\.\d+)"

| stats count by ip, username

| sort - count

The command retrieves OpenSSH logs, extracts failed SSH login attempts, and counts occurrences per IP and username. A ranked list helps detect brute-force attacks or intrusion attempts, enabling security teams to block suspicious IPs.

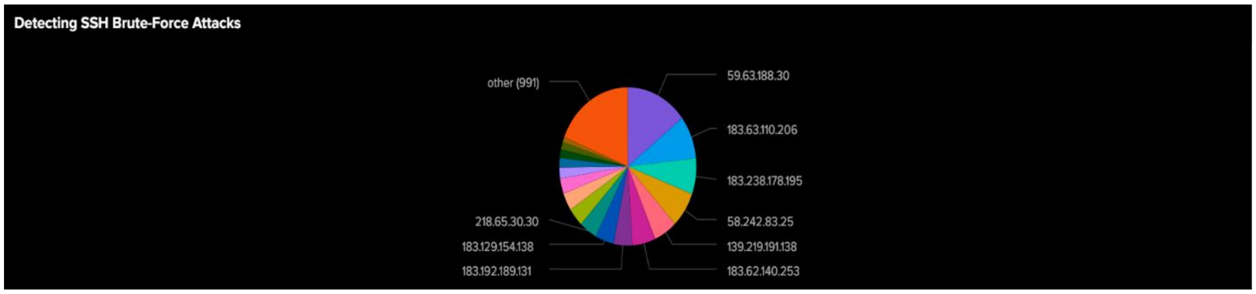


Figure 21: OpenSSH – SSH Brute-Force Attacks

5.2 Geolocation of Failed SSH Logins

Command: index=* source="OpenSSH server.log" host="openssh" sourcetype="openssh server"

| search "Failed password"

| rex field=_raw "Failed password for .* from (?<ip>\d+\.\d+\.\d+\.\d+)"

| iplocation ip

| geostats count by Country

The command retrieves OpenSSH logs, extracts failed SSH login attempts, and maps IPs to geographic locations. A choropleth map helps track SSH brute-force attack sources, enabling security teams to implement geo-based access restrictions.

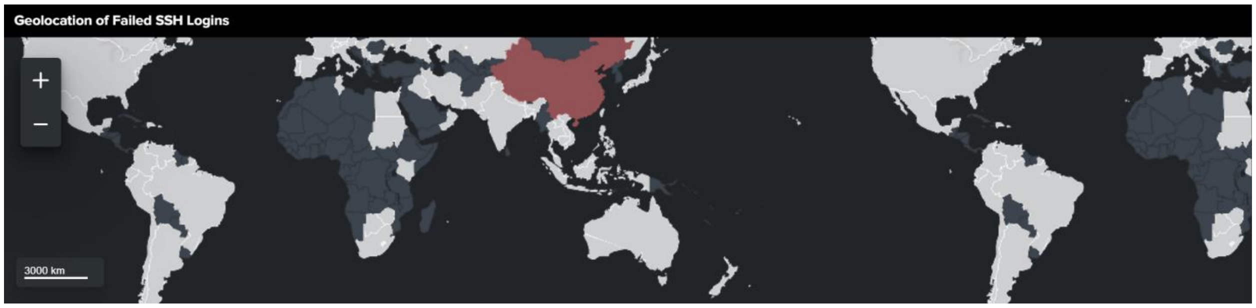


Figure 22: OpenSSH – Geolocation of Failed SSH Logins

5.3 Most Frequent Attack Sources

Command: index=* source="OpenSSH server.log" host="openssh" sourcetype="openssh server"

| search "Failed password"

| rex field=_raw "Failed password for .* from (?<ip>\d+\.\d+\.\d+\.\d+)"

| stats count by ip

| sort - count

| head 10

The command retrieves OpenSSH logs, extracts failed SSH login attempts, and identifies the top 10 attacking IPs. This helps detect repeat brute-force attackers, allowing security teams to block malicious IPs and enhance access controls.

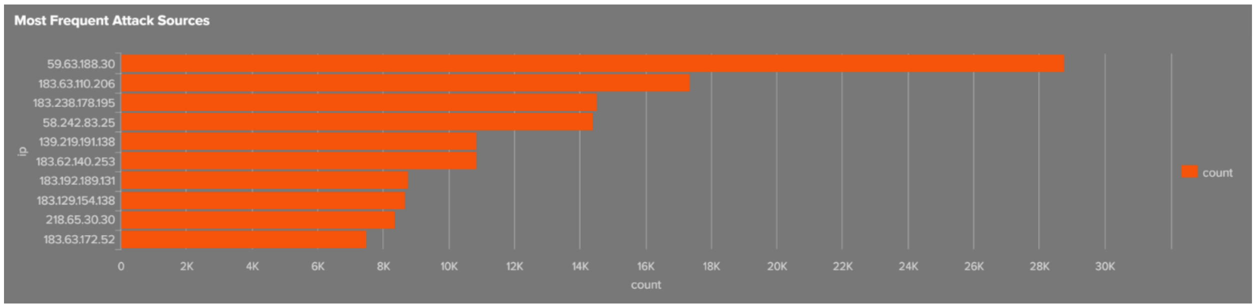


Figure 23: OpenSSH – Most Frequent Attack Sources

5.4 SSH Login Attempts Over Time

Command: index=* source="OpenSSH server.log" host="openssh" sourcetype="openssh server"

| search "Accepted password" OR "Failed password"

| timechart span=1h count by _raw

The command retrieves OpenSSH logs, tracks successful and failed SSH logins, and visualizes them hourly. A time-series graph helps detect brute-force attack trends, abnormal login spikes, and unauthorized access attempts, aiding threat detection.

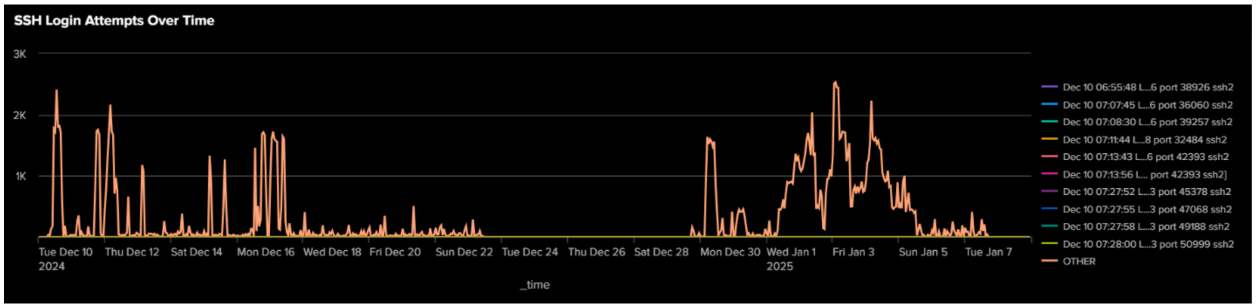


Figure 24: OpenSSH – SSH Login Attempts Over Time

5.5 Successful SSH Logins by User

Command: index=* source="OpenSSH server.log" host="openssh" sourcetype="openssh server"

| search "Accepted password"

| rex field=_raw "Accepted password for (?<username>\S+) from (?<ip>\d+\.\d+\.\d+\.\d+)"

| stats count by username

The command retrieves OpenSSH logs, extracts successful SSH logins, and counts occurrences per user. This helps track authorized access, detect anomalies in login frequency, and identify potential compromised accounts.

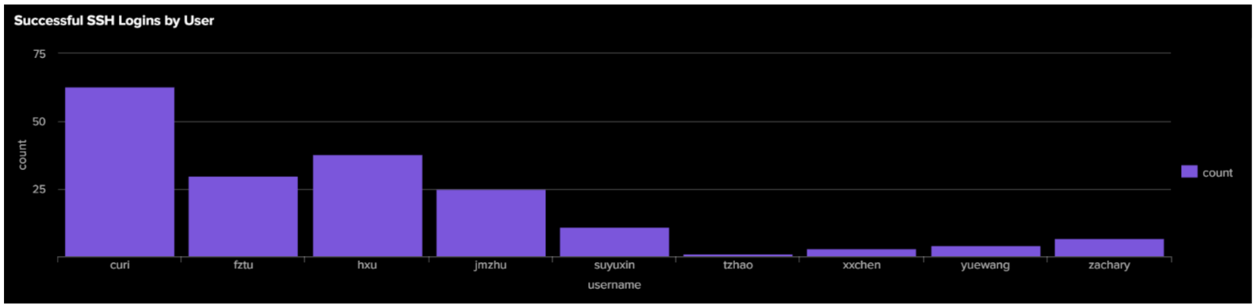


Figure 25: OpenSSH – Successful SSH Logins by User

4. Summary of Outcomes

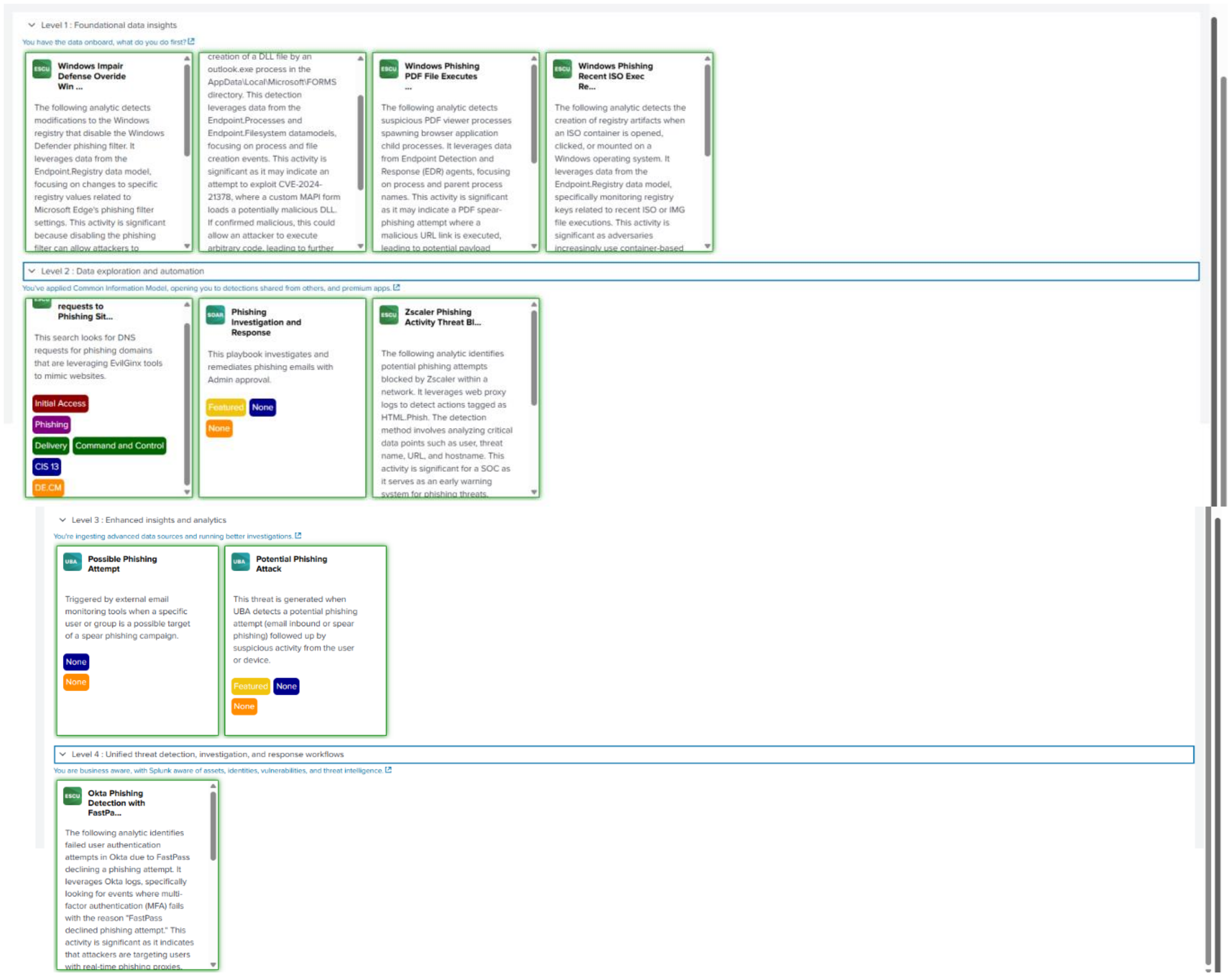
The deployment of Splunk within a SOC environment significantly improved threat detection and incident response. Security teams gained enhanced visibility, real-time alerting, and a strengthened security posture. The application of Splunk extends beyond security monitoring—it can be leveraged for IT operations, compliance reporting, fraud detection, and even business analytics. As organisations evolve, Splunk's machine learning and automation capabilities can further enhance threat detection, reduce response times, and streamline security workflows. The integration of AI-driven analytics ensures proactive identification of threats, making Splunk an invaluable tool in modern cybersecurity frameworks.

5. Wow Factor:

5.1 Splunk Security Essentials App

To enhance our SOC's capabilities, we integrated Splunk Security Essentials, a tool that provides out-of-the-box security analytics. This enabled us to:

- Automate Phishing Detection using DNS lookups, email analysis, and anomaly detection.
- Enhance Threat Investigation with built-in playbooks for phishing detection and response.
- Monitor User Authentication by detecting credential compromises and login anomalies.
- Improve Threat Response Workflows through real-time alerts and preconfigured dashboards.
- By implementing Splunk Security Essentials, we transformed raw log data into actionable security insights, enabling faster threat detection, automated responses, and proactive security measures.



Level 1 : Foundational data insights

You have the data onboard, what do you do first?

Windows Impair Defense Overide Win ...

ESCU

The following analytic detects modifications to the Windows registry that disable the Windows Defender phishing filter. It leverages data from the Endpoint.Registry data model, focusing on changes to specific registry values related to Microsoft Edge's phishing filter settings. This activity is significant because disabling the phishing filter can allow attackers to

creation of a DLL file by an outlook.exe process in the AppData\Local\Microsoft\FORMS directory. This detection leverages data from the Endpoint.Processes and Endpoint.Filesystem datamodels, focusing on process and file creation events. This activity is significant as it may indicate an attempt to exploit CVE-2024-21378, where a custom MAPI form loads a potentially malicious DLL. If confirmed malicious, this could allow an attacker to execute arbitrary code, leading to further

Windows Phishing PDF File Executes ...

ESCU

The following analytic detects suspicious PDF viewer processes spawning browser application child processes. It leverages data from Endpoint Detection and Response (EDR) agents, focusing on process and parent process names. This activity is significant as it may indicate a PDF spear-phishing attempt where a malicious URL link is executed, leading to potential payload

Windows Phishing Recent ISO Exec Re...

ESCU

The following analytic detects the creation of registry artifacts when an ISO container is opened, clicked, or mounted on a Windows operating system. It leverages data from the Endpoint.Registry data model, specifically monitoring registry keys related to recent ISO or IMG file executions. This activity is significant as adversaries increasingly use container-based

Level 2 : Data exploration and automation

You've applied Common Information Model, opening you to detections shared from others, and premium apps.

requests to Phishing Sit...

This search looks for DNS requests for phishing domains that are leveraging EvilGinx tools to mimic websites.

Initial Access

Phishing

Delivery Command and Control

CIS 13

DE.CM

Phishing Investigation and Response

SOAR

This playbook investigates and remediates phishing emails with Admin approval.

Featured None

None

Zscaler Phishing Activity Threat Bl...

ESCU

The following analytic identifies potential phishing attempts blocked by Zscaler within a network. It leverages web proxy logs to detect actions tagged as HTML.Phish. The detection method involves analyzing critical data points such as user, threat name, URL, and hostname. This activity is significant for a SOC as it serves as an early warning system for phishing threats.

Level 3 : Enhanced insights and analytics

You're ingesting advanced data sources and running better investigations.

Possible Phishing Attempt

UBA

Triggered by external email monitoring tools when a specific user or group is a possible target of a spear phishing campaign.

None

None

Potential Phishing Attack

UBA

This threat is generated when UBA detects a potential phishing attempt (email inbound or spear phishing) followed up by suspicious activity from the user or device.

Featured None

None

Level 4 : Unified threat detection, investigation, and response workflows

You are business aware, with Splunk aware of assets, identities, vulnerabilities, and threat intelligence.

Okta Phishing Detection with FastPa...

ESCU

The following analytic identifies failed user authentication attempts in Okta due to FastPass declining a phishing attempt. It leverages Okta logs, specifically looking for events where multi-factor authentication (MFA) fails with the reason "FastPass declined phishing attempt." This activity is significant as it indicates that attackers are targeting users with real-time phishing proxies.

Figure 26: Splunk Security Essentials Overview

5.2 Integration of Cyber Security Essentials App in Splunk

To enhance the capabilities of my Splunk deployment, I installed and explored the Cyber Security Essentials app, available from the Splunkbase. This app adds advanced use cases, dashboards, and detection playbooks that simulate how a Security Operations Centre (SOC) would monitor real-time threats. Although no live threats were detected during this demonstration, the tool visually represents the architecture of a modern cyber threat detection system.

The Cyber Security Essentials dashboard includes panels such as:

- All Threats

- APT Actors
- Malware and Exploits
- Triggered Detections
- Detection Rules (Enabled vs Disabled)

This integration demonstrates the extensibility of Splunk through add-ons and enhances its value beyond basic log analysis. It serves as a great learning resource and showcases the potential for enterprise-level use.

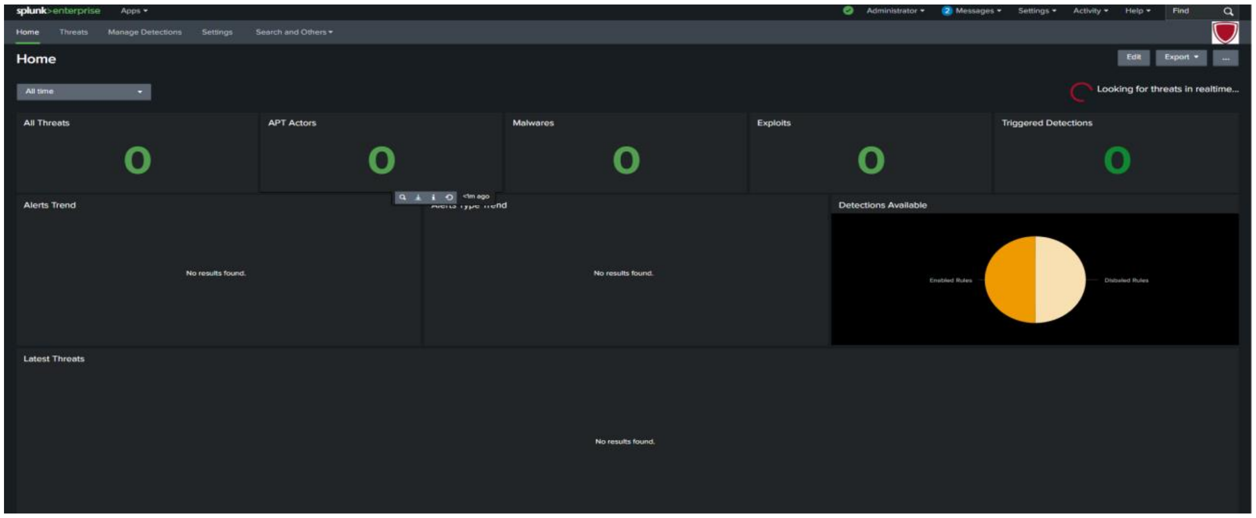


Figure 27: Cyber Security Essentials Dashboard

5.3 System Monitoring for my operating system (Windows 11)

This Splunk query monitors system restarts, shutdowns, and planned restarts by retrieving relevant logs from the Windows System Event Log. The search filters for three specific event codes: 6005 (System Started), 6006 (System Shutdown), and 1074 (Planned Restart). To enhance readability, the eval function assigns user-friendly labels to each event type. Finally, the timechart command visualizes the count of these events over time, grouped into hourly intervals. The best visualization for this data is a line chart, which effectively tracks system activity trends, helping to detect unexpected shutdowns, frequent restarts, or potential system stability issues.

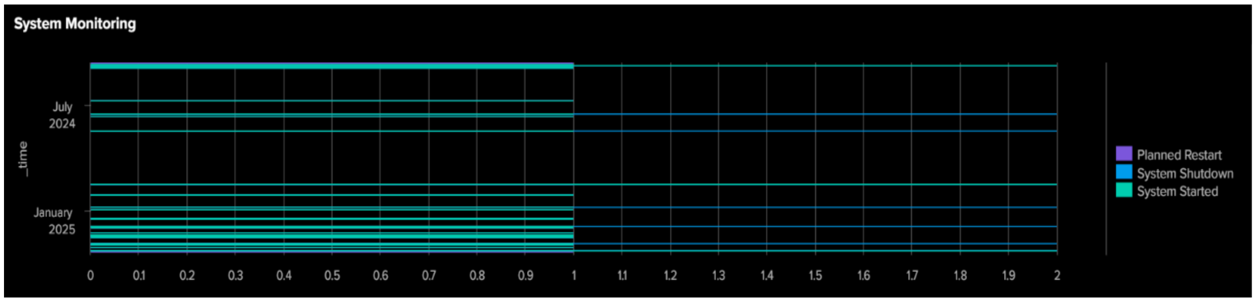


Figure 28: Windows 11 System Monitoring Panel

6. Conclusion

This project demonstrated how Splunk can be used effectively to monitor and analyse logs across various systems. Through simple SPL queries and dashboards, important patterns such as user activity, traffic behaviour, and potential security issues were identified. The addition of apps like Cyber Security Essentials highlighted how Splunk can be extended for more advanced security use cases. Overall, the deployment showed that Splunk is a practical and powerful tool for supporting SOC operations.

7. References

1 . Pan, X. (2024). Independent Study of Splunk. *OAlib*, 11(04), pp.1–16. doi:https://doi.org/10.4236/oalib.1111496

2. Balaji, N., Karthik Pai, B.H., Bhat, B. and Praveen, B. (2021). Data Visualization in Splunk and Tableau: A Case Study Demonstration. *Journal of Physics: Conference Series*, 1767(1), p.012008. doi:https://doi.org/10.1088/1742-6596/1767/1/012008.

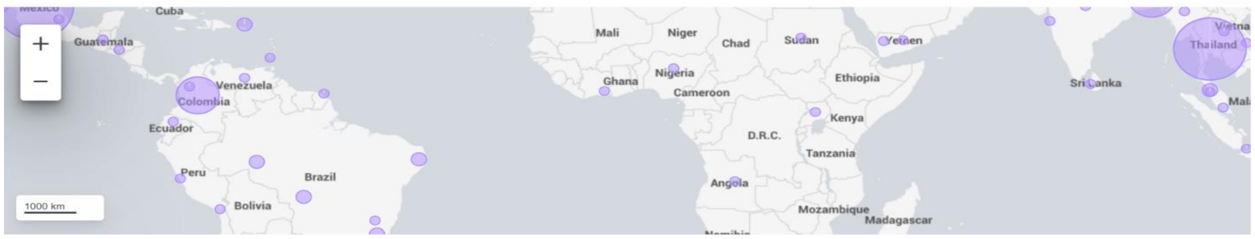
8.Appendix:

Video Demo Link:

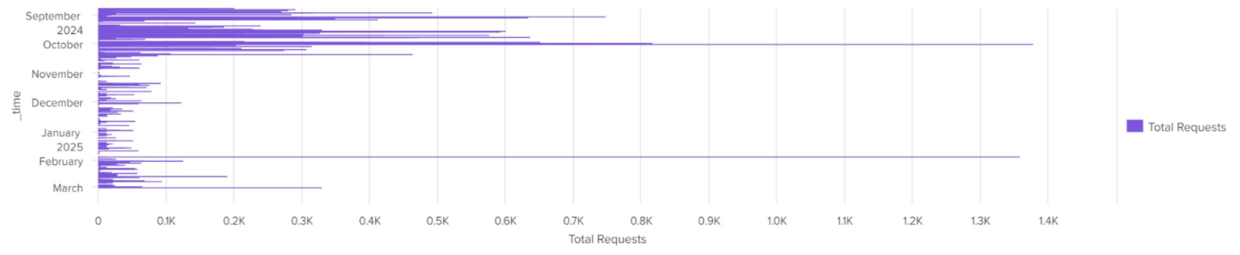
https://kingstonuniversity-my.sharepoint.com/:v:/g/personal/k2152095_kingston_ac_uk/ETM2o-yd9qlCjk81zBNVve8BY5CYs6gvllwZoFh6z56Vlw?e=1y1Bhh

Dashboard 1: Apache web server log

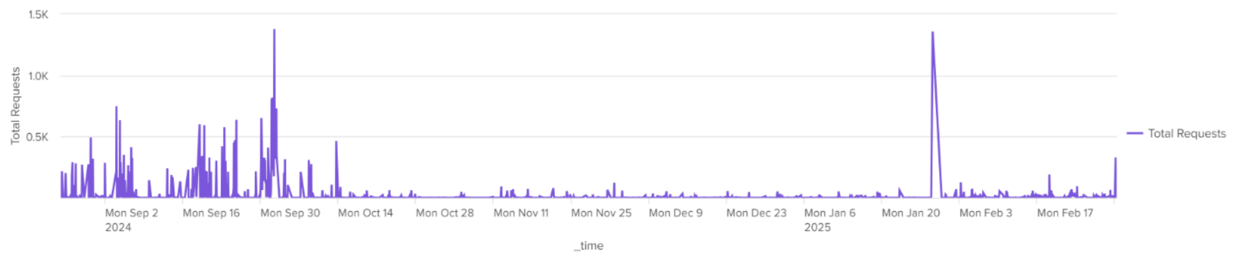
Countries with the Most Requests



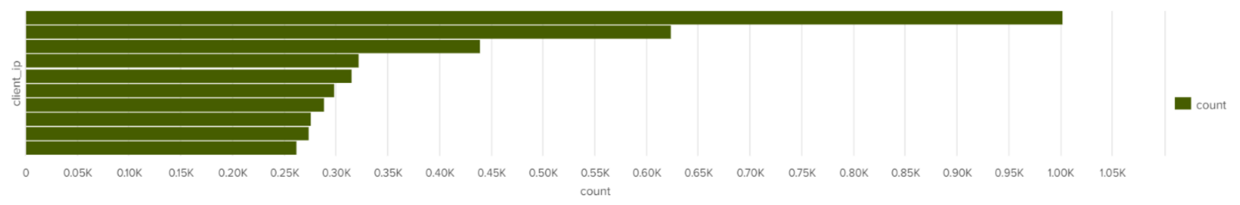
Peak Traffic Hours Panel



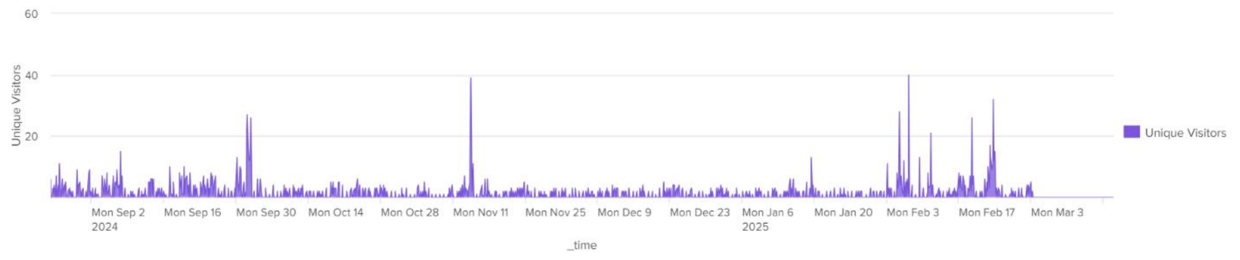
Requests Over Time



Top 10 Client IPs Making Requests

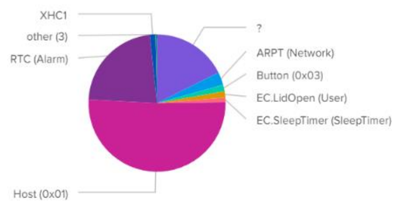


Unique Visitors Over Time

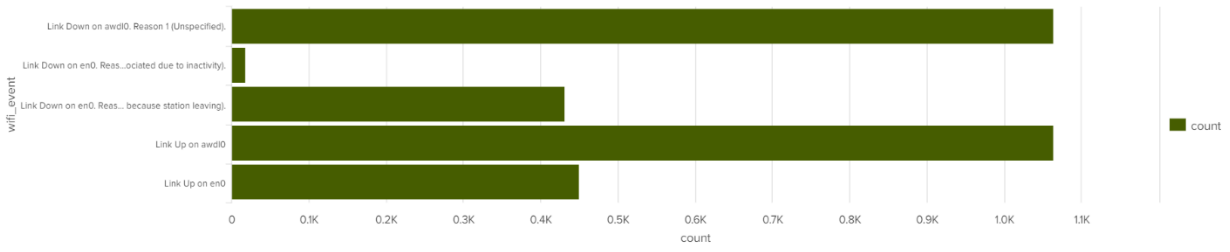


Dashboard 2 : Mac OS log

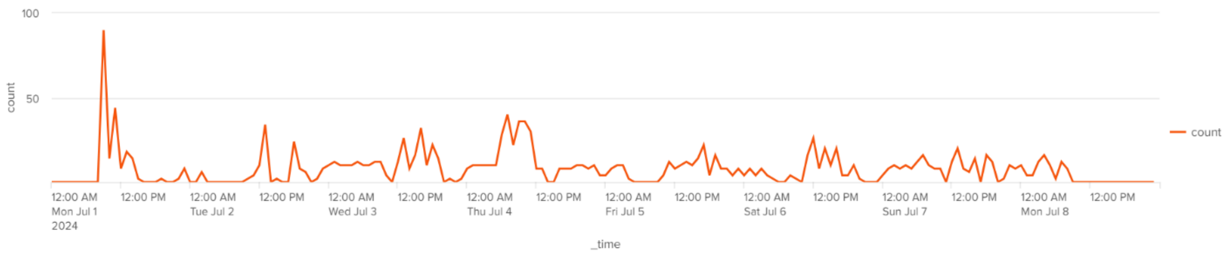
System Wake Reasons



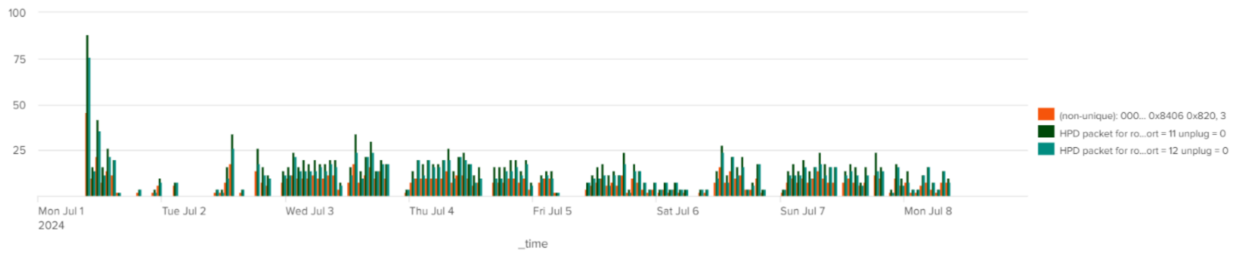
Wi-Fi Connection & Disconnection Logs



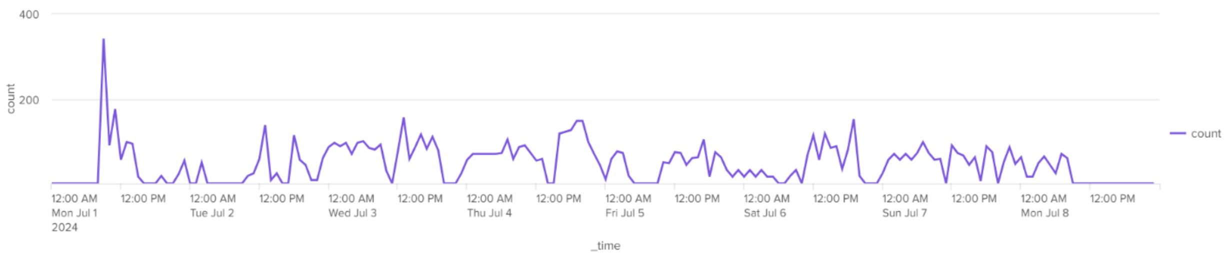
Wi-Fi Connection Failures Over Time



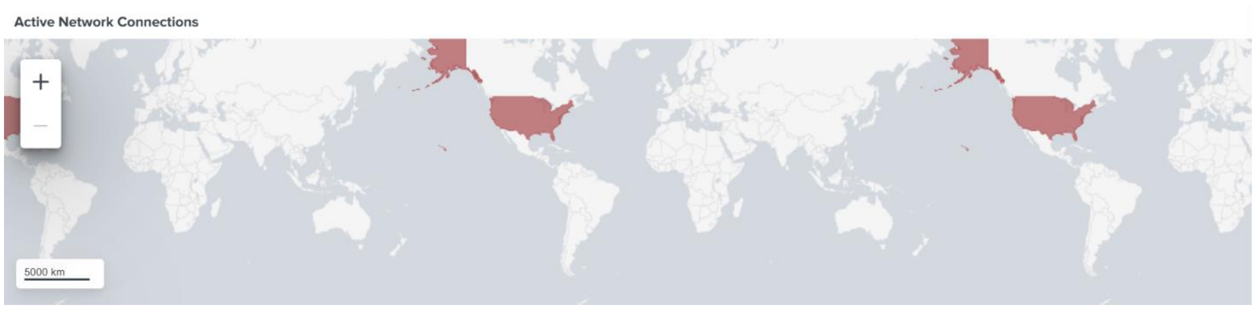
USB & Thunderbolt Device Trends

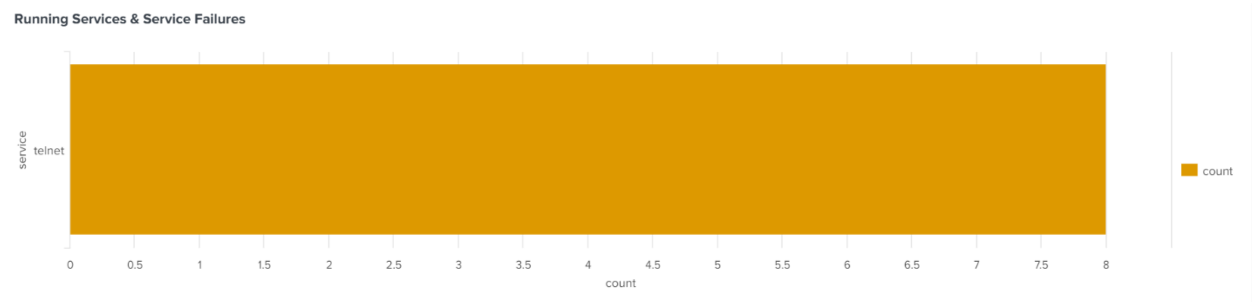


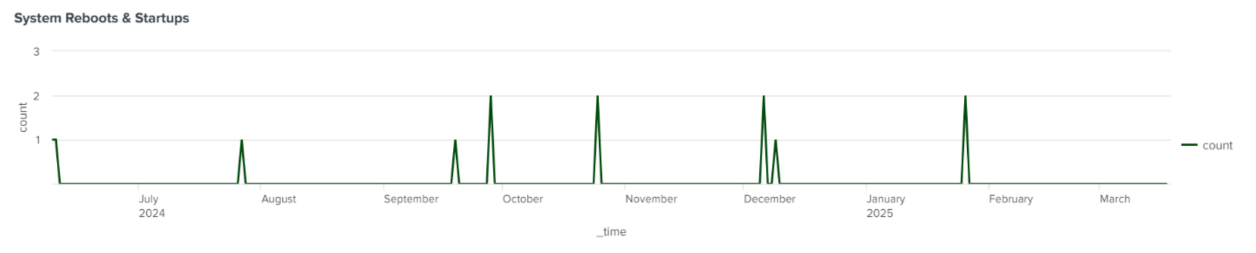
System Camera Activity

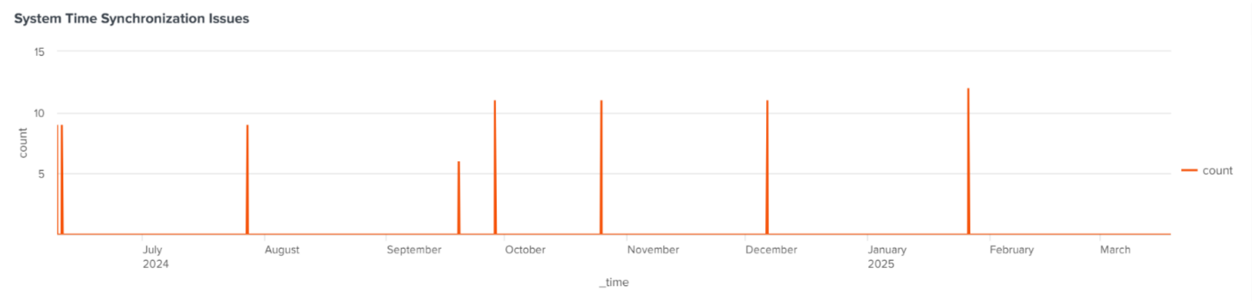


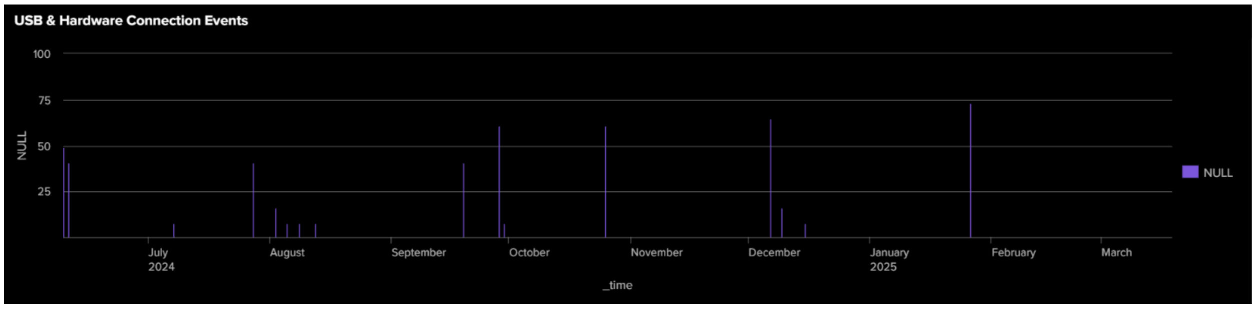
Dashboard 3: Linux system log





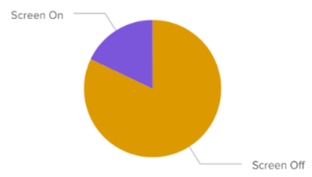




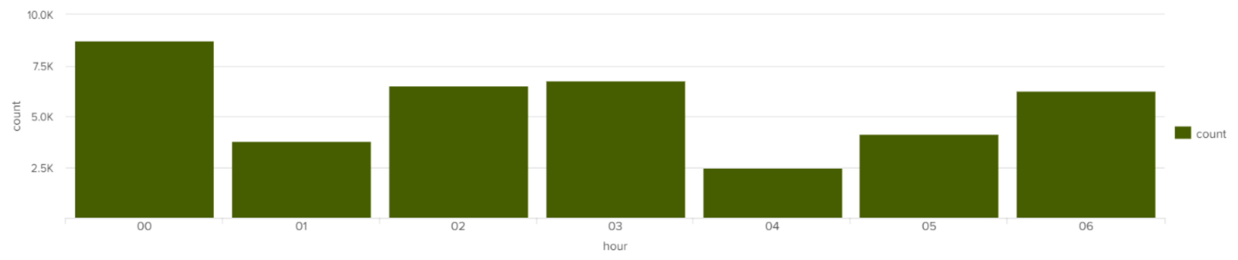


Dashboard 4: Health App Log

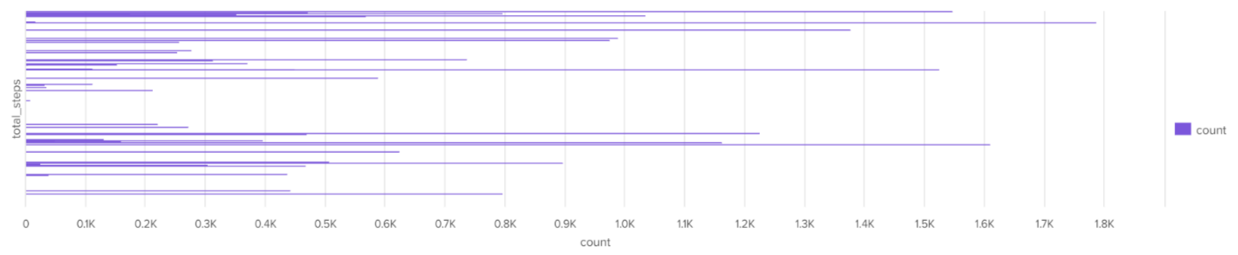
Correlating Steps & Screen Activity



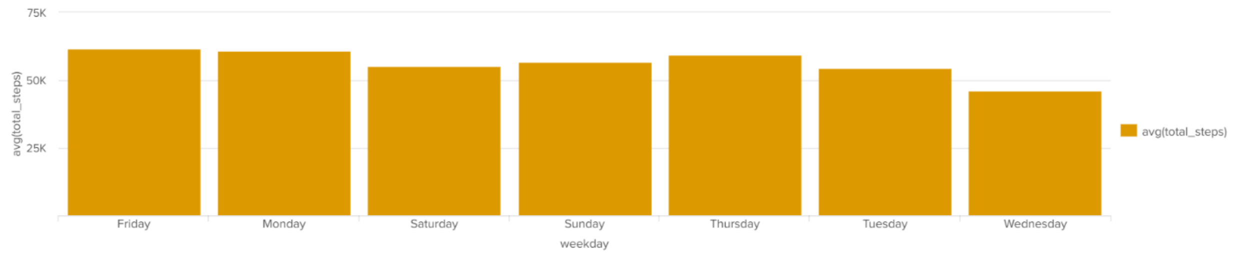
Detecting Unusual Activity During Midnight Hours



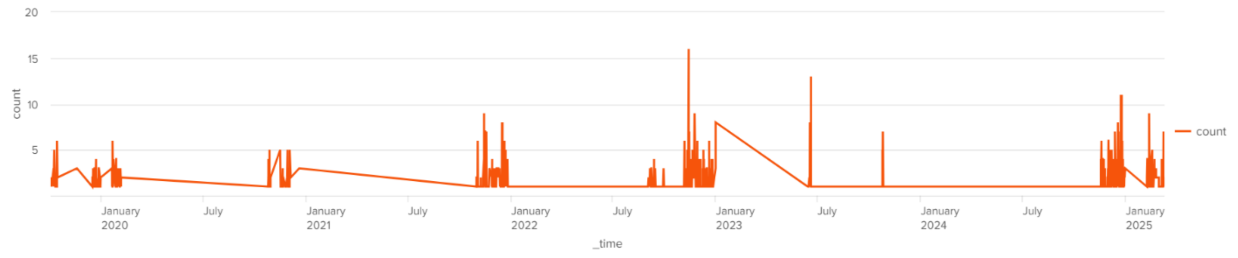
Identifying Data Integrity Issues



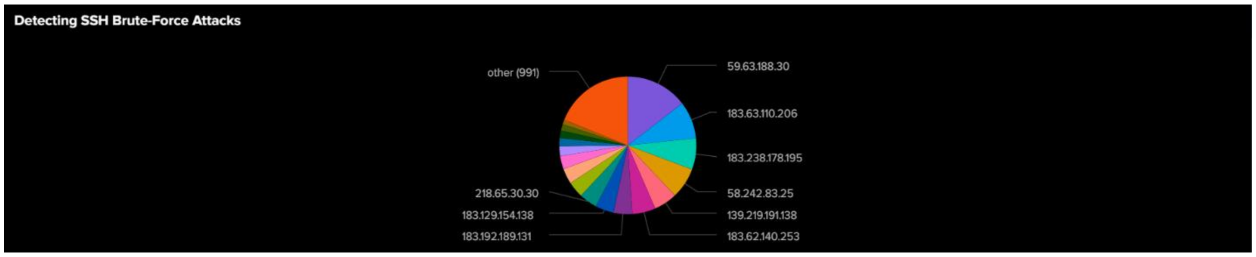
Monitoring User Activity by Weekday



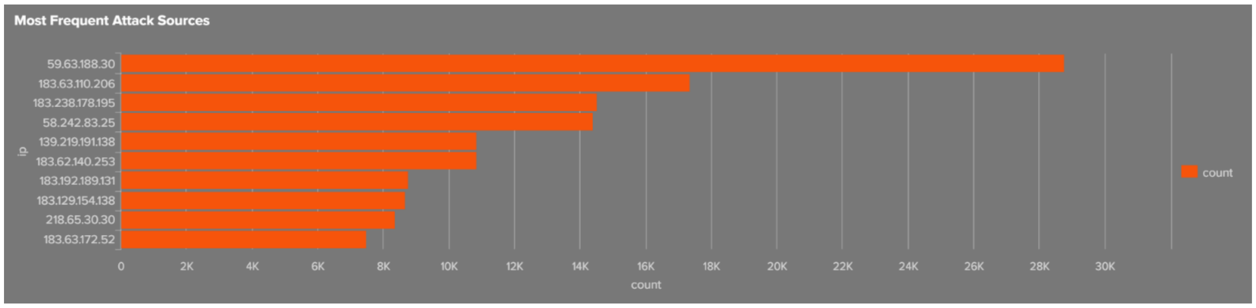
Detecting Frequent Screen Locking

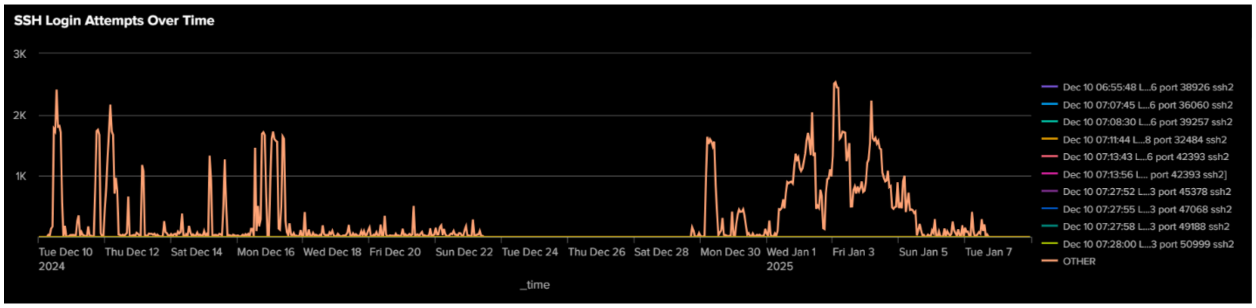


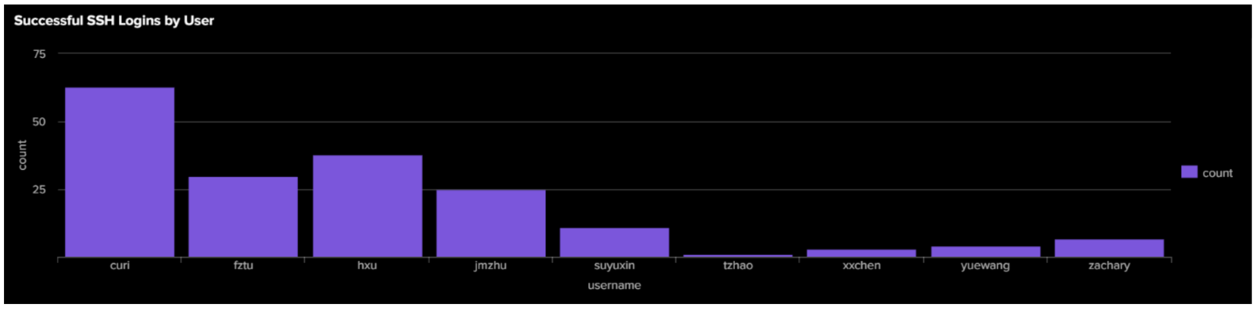
Dashboard 5: OpenSSH server log



Geolocation of Failed SSH Logins
+
−
3000 km


Most Frequent Attack Sources
ip
2K
4K
6K
8K
10K
12K
14K
16K
18K
20K
22K
24K
26K
28K
30K
count
count


SSH Login Attempts Over Time
3K
2K
1K
Tue Dec 10 2024
Thu Dec 12
Sat Dec 14
Mon Dec 16
Wed Dec 18
Fri Dec 20
Sun Dec 22
Tue Dec 24
Thu Dec 26
Sat Dec 28
Mon Dec 30
Wed Jan 1 2025
Fri Jan 3
Sun Jan 5
Tue Jan 7
_time
Dec 10 06:55:48 L...6 port 38926 ssh2
Dec 10 07:07:45 L...6 port 36060 ssh2
Dec 10 07:08:30 L...6 port 39257 ssh2
Dec 10 07:11:44 L...8 port 32484 ssh2
Dec 10 07:13:43 L...6 port 42393 ssh2
Dec 10 07:13:56 L... port 42393 ssh2]
Dec 10 07:27:52 L...3 port 45378 ssh2
Dec 10 07:27:55 L...3 port 47068 ssh2
Dec 10 07:27:58 L...3 port 49188 ssh2
Dec 10 07:28:00 L...3 port 50999 ssh2
OTHER


Successful SSH Logins by User
count
curi
fztu
hxu
jmzhu
suyuxin
tzhao
xxchen
yuewang
zachary
username
count