Defensive Security Project
Splunk SIEM Emulation

Table of Contents

This document contains the following resources:

Monitoring Environment

Scenario

- We are SOC analysts at Virtual Space Industries (VSI), a company designing virtual-reality programs for businesses.
- VSI is worried that a competitor, JobeCorp, may launch cyberattacks to disrupt operations.
- Our role: Use Splunk to monitor and detect potential threats to VSI’s systems and applications.
- Key assets to monitor:
- Apache web server for the admin webpage.
  Windows OS handling critical back-end functions.
- Past logs from the networking team are available to help us establish baselines of regular expected activity.
- Logs provided:
- Windows Server Logs with intellectual property on VR projects.
- Apache Server Logs for the public website, vsi-company.com.
  - We will use the provided logs to build a comprehensive set of reports, alerts, and dashboards in Splunk.
  - These tools will help us proactively identify and mitigate any potential cyberattacks from JobeCorp or other bad actors.

Add-On for DNS Lookup

Add-On for Splunk: DNS Lookup

- Enhance your Splunk searches with DNS query results for various record types from any DNS server, adding context to your data analysis.
- Supported lookups include common DNS records like A, MX, NS, and TXT, helping you investigate IP addresses, mail servers, and domain aliases efficiently.
- Reverse lookups are available to resolve IP addresses back to hostnames, aiding in network investigations.
- The tool offers pre-configured lookups using popular DNS resolvers like Google, CloudFlare, and OpenDNS, making it flexible and easy to use.
- Users can customize these DNS lookups to suit their needs, including using specific DNS servers or creating new query types.
- Commercial support is available for all apps, ensuring professional assistance if needed.
- Additional use cases and insights are shared on the DNS-Based Threat Intelligence blog, providing real-world applications of the tool.

Add-On for Splunk: DNS Lookup

Scenario:

Imagine we're monitoring network traffic and notice one of the top 10 referrer domains looks suspicious. Without more context, it's hard to tell if this domain is a real threat.

By using a DNS lookup tool add-on for Splunk, we can get detailed information about the domain, such as who registered it, its associated IP addresses, and whether it has a history of malicious activity. This extra data helps us quickly decide if the domain is dangerous and take action if needed. Additionally, the tool can fetch email-related information, like which servers are authorized to send emails for a domain, making it easier to detect potential phishing attacks.

It can also give us insights into how a domain's online infrastructure is set up, which could reveal vulnerabilities or misconfigurations. Overall, this tool enhances our ability to investigate suspicious activity, make better security decisions, and improve our defenses.

Add-On for DNS Lookup

Logs Analyzed

1 Windows Logs

The Windows logs for Splunk contain key fields:

- **signature_id**: Identifies the specific event or threat.
- **signature**: Describes the nature of the event.
- **user**: Indicates which user was involved.
- **status**: Shows the current state or outcome of the event.
- **severity**: Rates the event's impact or urgency

2 Apache Logs

The Apache logs for Splunk contain essential fields:

- **method**: Type of HTTP request made (e.g., GET, POST).
- **referer_domain**: Domain that referred the request to our server.
- **status**: HTTP response code indicating the request outcome.
- **clientip**: IP address of the client making the request.
- **useragent**: Information about the client's browser or device.

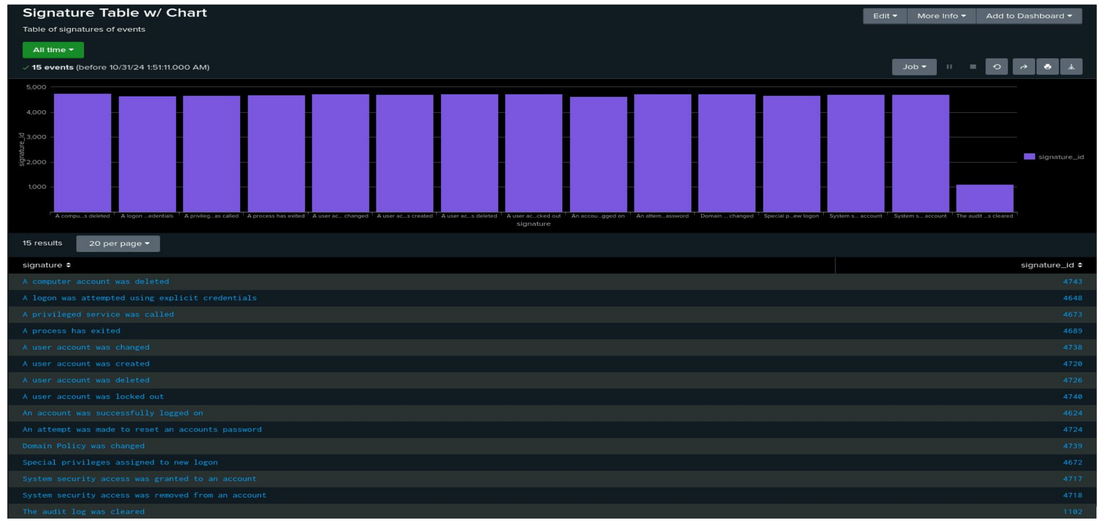
Windows Logs

Reports—Windows

Designed the following reports:

Report Name	Report Description
Signature & Signature IDs	Filters events and removes duplicate entries based on the **signature** and **signature_id** fields, displays these fields in a table, and then sorts the table by the **signature** field.
Alert Severity by Percentage	Filters events by **severity**, counts the number of events for each severity level. It then calculates the total event count and determines the percentage of each severity level relative to the total. The results are displayed in a table showing **severity**, **count**, and **percentage**, sorted by the event count in descending order.
Successes & Failures	Filters events with a **status** of either "**success**" or "**failure**." It calculates the total count, and determines the percentage for each status. Data is then displayed in a table showing the **status**, **count**, and **percentage**, sorting the table in descending order.

Images of Reports—Windows



Alerts—Windows

Designed the following alerts:

Alert Name	Alert Description	Alert Baseline	Alert Threshold
Hourly Failed Activities	Filters for events with **status** set to "**failure**" by one-hour intervals, and counts the number of failures for each hour. Creates a time chart displaying total failures per hour.	5.92	> 10

JUSTIFICATION: Used SPL Queries: ‘status="failure" | timechart span=1h count as Failed_Attempts | stats avg(Failed_Attempts) as Avg_Failed_Attempts’ & ‘status="failure" | timechart span=1h count as Failed_Attempts | where Failed_Attempts > 6’ to calculate the hourly failure rate, which averaged 5.92, and identify peak failures, reaching 10 in a single one-hour interval.

Alerts—Windows

Designed the following alerts:

Alert Name	Alert Description	Alert Baseline	Alert Threshold
Hourly User Account Logins	Filters for events with the **signature** "**an account was successfully logged on**," by one-hour intervals, and counts occurrences grouped by **signature_id**. Creates a time chart displaying the number of successful logons per hour for each **signature_id.**	13.46	> 21

JUSTIFICATION: Used SPL Queries: ‘signature_id="4624" | timechart span=1h count as Successful_Logins | stats avg(Successful_Logins) as Avg_Successful_Logins’ & ‘signature_id="4624" | timechart span=1h count as Successful_Logins | where Successful_Logins > 14’ to calculate the hourly successful login rate, which averaged 13.46, and identify peak failures, reaching 21 in a single one-hour interval.

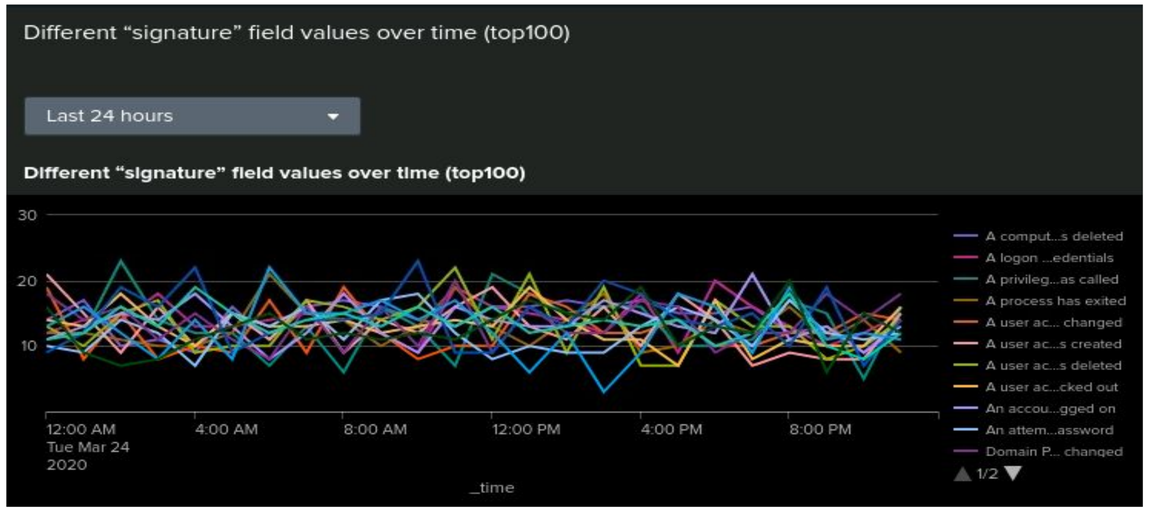
Alerts—Windows

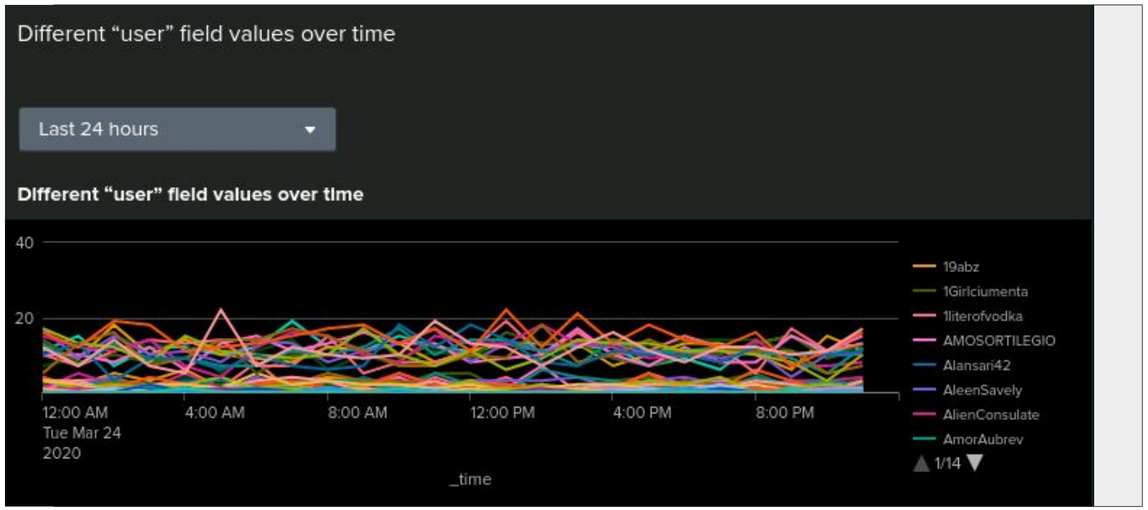
Designed the following alerts:

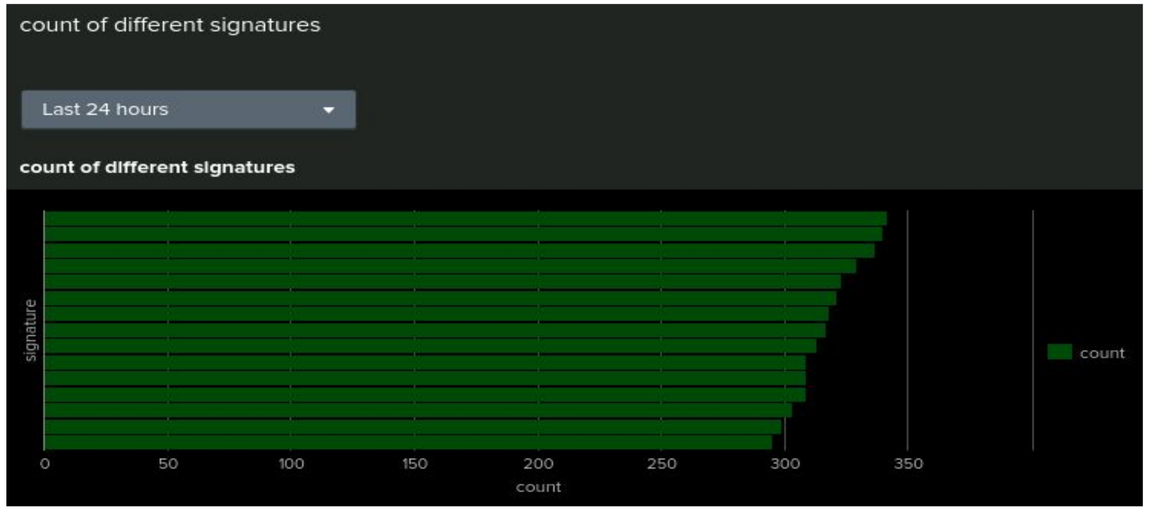
Alert Name	Alert Description	Alert Baseline	Alert Threshold
Hourly User Account Deletion	Filters for events with **signature_id** "**4726**," which relates to user account deletions. It organizes the events into one-hour intervals and counts the deletions per hour. The results are displayed in a time chart.	13.25	> 22

JUSTIFICATION: Used SPL Queries: ‘signature_id="4726" | timechart span=1h count as User_Account_Deleted | stats avg(User_Account_Deleted) as Avg_Deleted’ & ‘signature_id="4726" | timechart span=1h count as User_Account_Deleted | where User_Account_Deleted > 14’ to calculate the hourly user account deletion rate, which averaged 13.25, and identify peak failures, reaching 22 in a single one-hour interval.

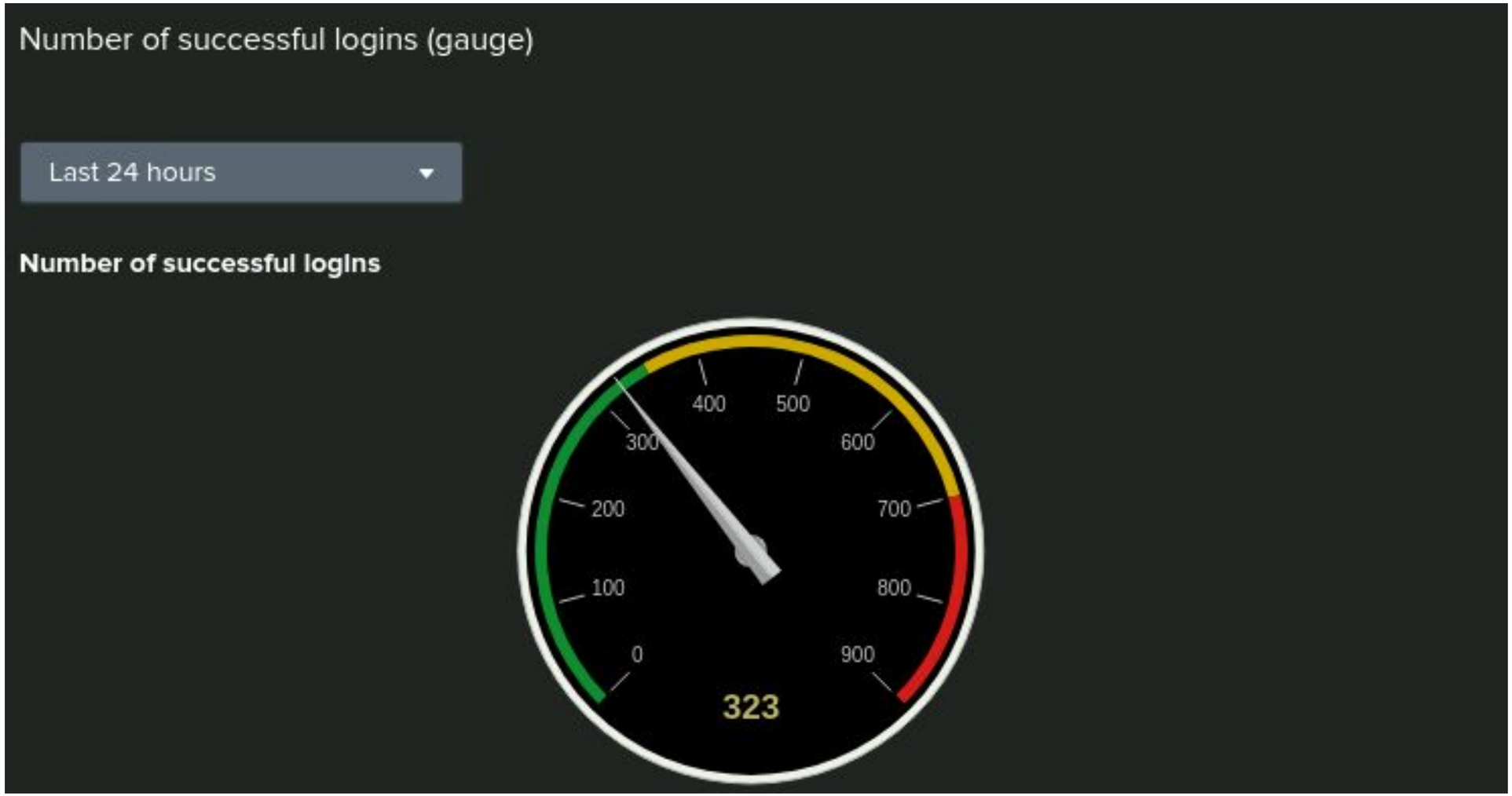
Dashboards—Windows







Dashboards—Windows



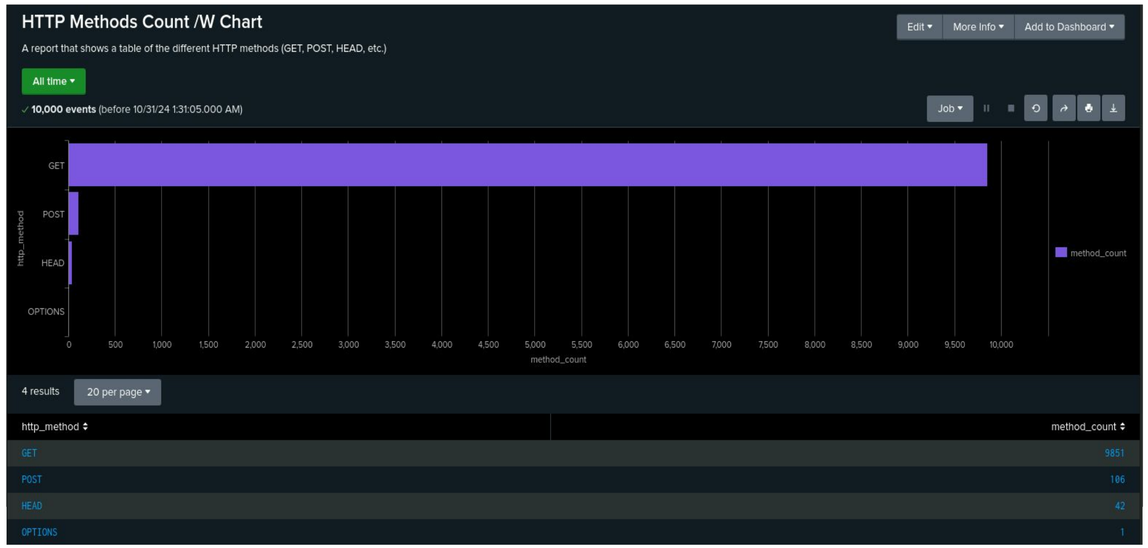
Apache Logs

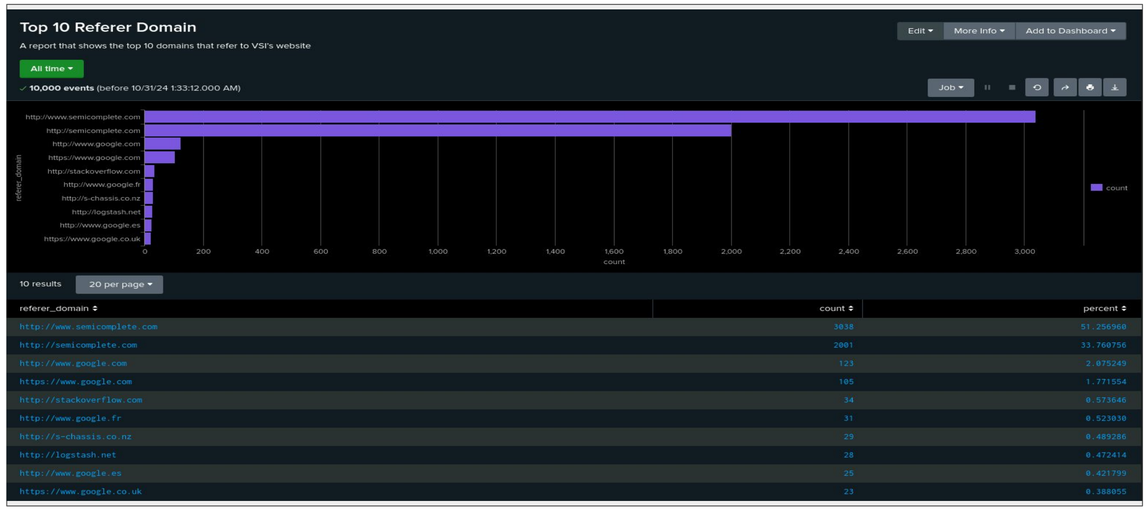
Reports—Apache

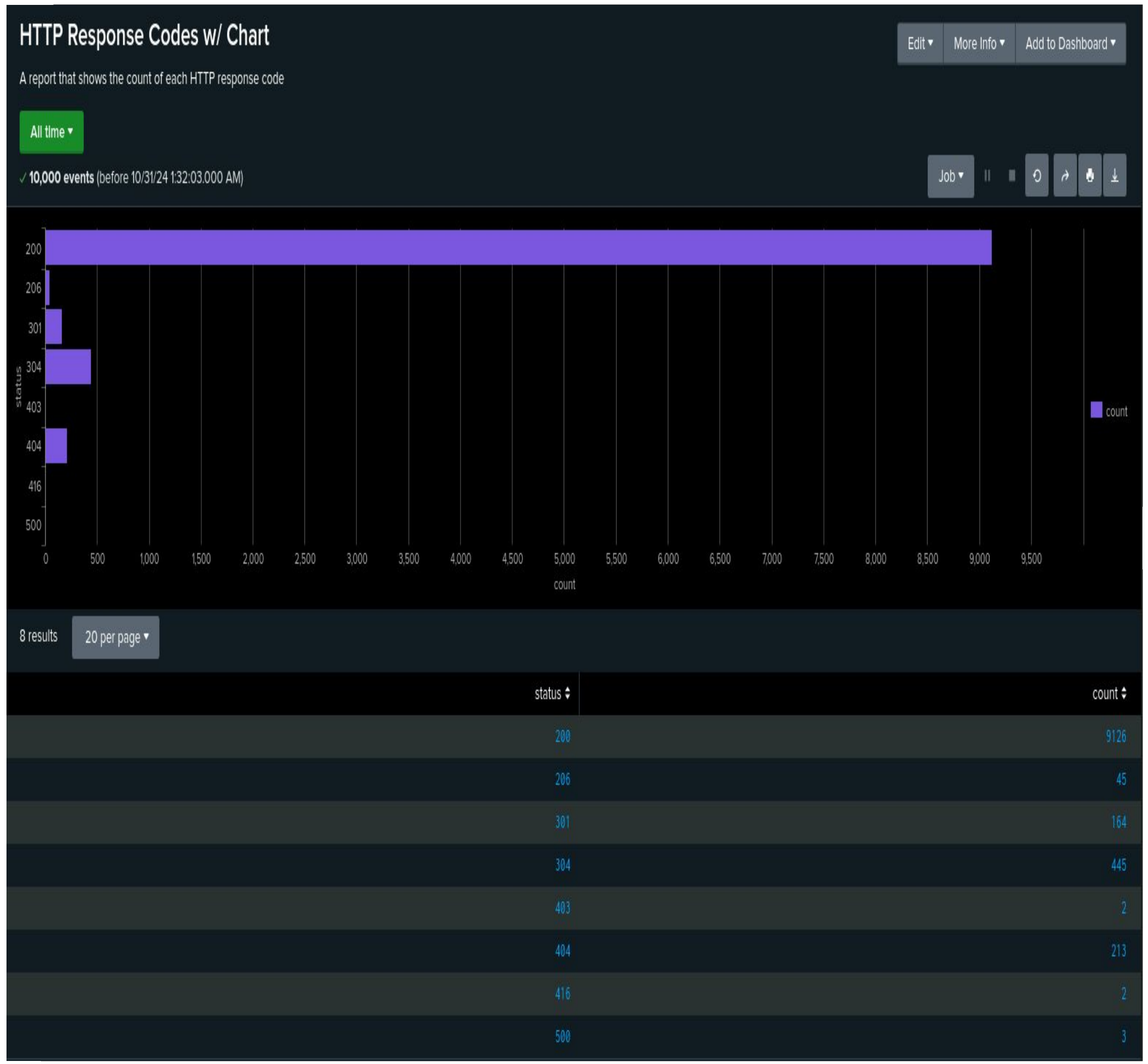
Designed the following reports:

Report Name	Report Description
HTTP Methods	Creates a table based on the value of the **method** field, categorizing it into common HTTP methods or labeling it as "Other" if it doesn't match. It then counts the total of each HTTP **method** and sorts the results in descending order by **method_count**.
Top 10 Referrer Domains	Retrieves the top 10 most common values in the **referer_domain** field, showing the frequency and percentage of each. It provides a quick overview of the main domains referring traffic to **vsi-company.com**.
HTTP Response Codes	Counts the number of events for each unique HTTP **status** code. It gives an overview of the distribution of HTTP responses, such as successful requests (ex., 200) or errors (ex., 404), helping to identify issues in web traffic or server performance.

Images of Reports—Apache







Alerts—Apache

Designed the following alerts:

Alert Name	Alert Description	Alert Baseline	Alert Threshold
Hourly Activity From Non-US Countries	Uses **iplocation** to get location details from the **clientip** field and filters out events originating from the United States. It organizes the data into one-hour intervals and counts the number of events per hour.	73.10	> 120

JUSTIFICATION: Used SPL Queries: '| iplocation clientip | where Country != "United States" | bin _time span=1h | stats count by _time | eventstats avg(count) as average' & '| iplocation clientip | where Country!="United States" | bin _time span=1h |stats count by _time | where count > 75' to calculate the hourly amount of non-US based connections, which averaged 73.10, and identify the peak connection rate, 120 in an one-hour interval.

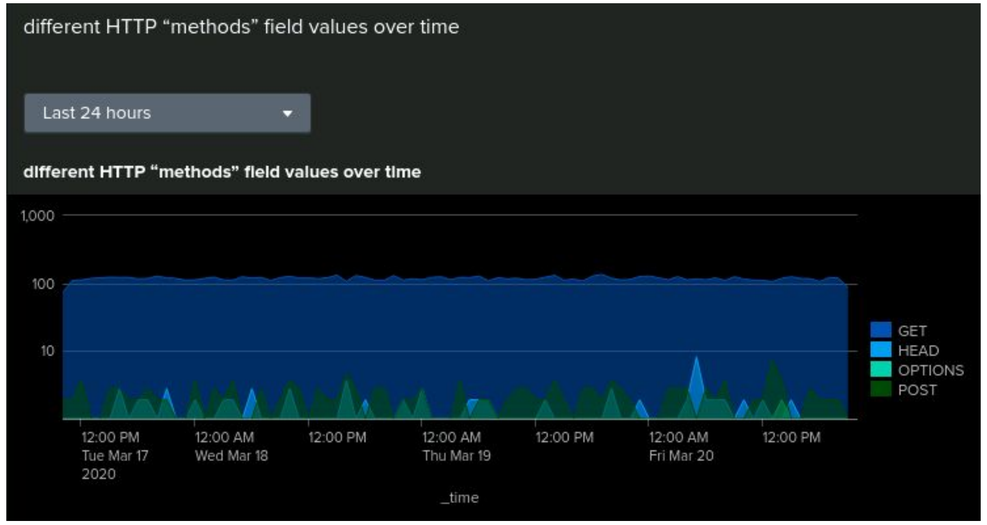
Alerts—Apache

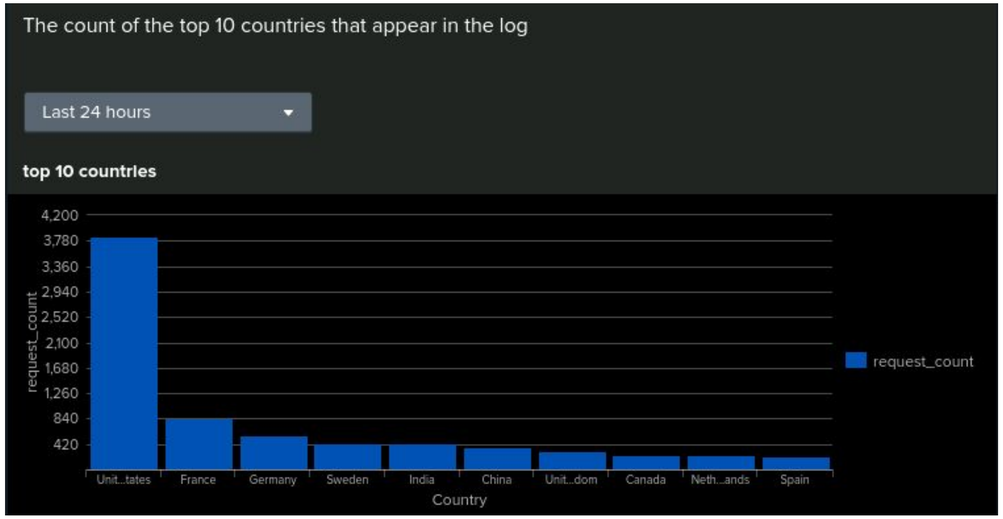
Designed the following alerts:

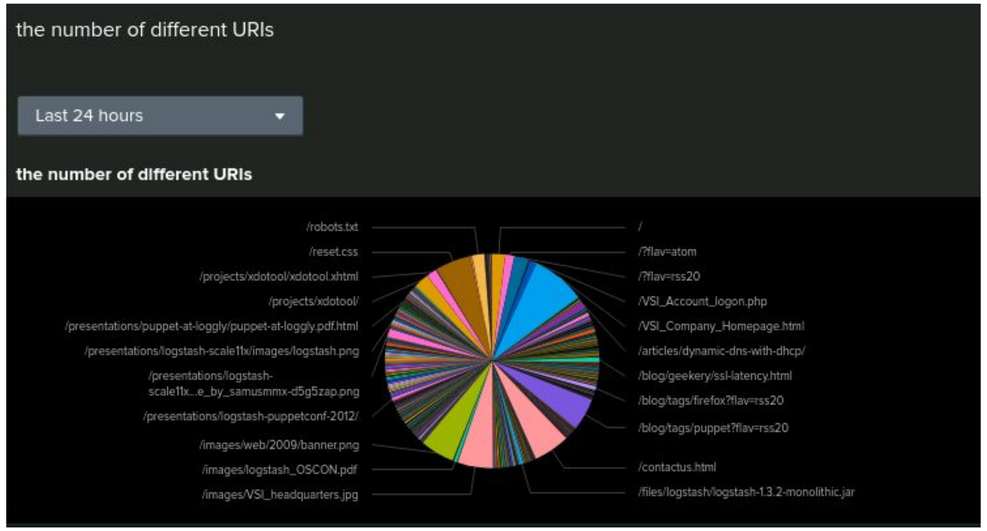
Alert Name	Alert Description	Alert Baseline	Alert Threshold
Hourly HTTP POST Methods	Filters for events where the **method** is "**POST**," organizes them into one-hour intervals, and counts the number of **POST** requests per hour	1.83	> 7

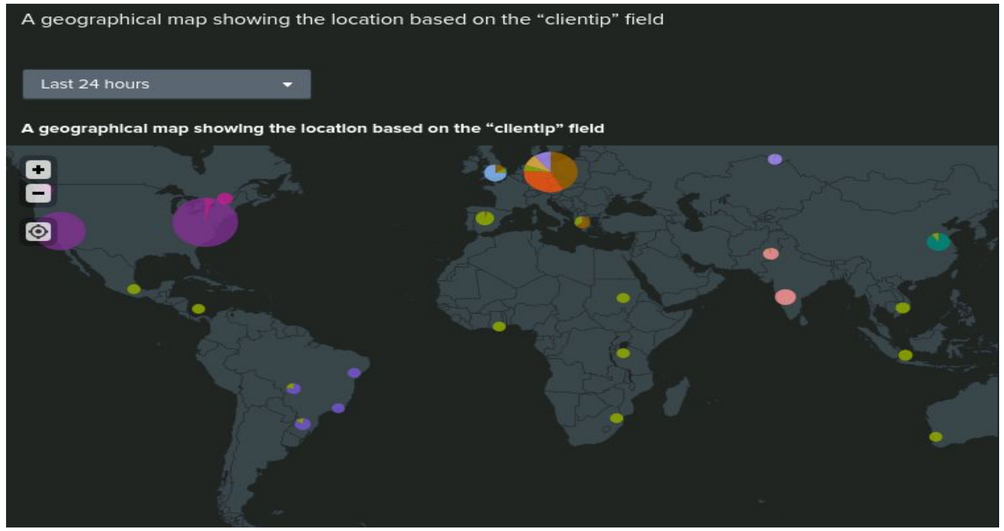
JUSTIFICATION: Used SPL Queries: ‘method="POST" | bin _time span=1h | stats count as post_count by _time | eventstats avg(post_count) as average’ & ‘method="POST" | bin _time span=1h | stats count as post_count by _time | where post_count > 2’ to calculate the hourly HTTP **POST methods**, which averaged 1.83, and identify the peak rate, which reached 7 in a one-hour interval.

Dashboards—Apache









Attack Analysis

Attack Summary—Windows

Summarize your findings from your reports when analyzing the attack logs.

- Our analysis of the attack logs revealed that only one out of our three alerts would have successfully detected the attackers.
  Specifically, the Windows Failed Activity alert would have triggered since the threshold was exceeded.
- However, the thresholds for the Hourly User Account Logins and Hourly User Account Deletions alerts were not surpassed. The attackers' actions stayed within the expected baseline for hourly activity, indicating that our current alert setup has limitations.
- This highlights a significant weakness: relying solely on threshold-based alerts can be ineffective if attackers operate within normal behavioral patterns, underscoring the need for more sophisticated detection methods.

Attack Summary—Windows

Summarize your findings from your alerts when analyzing the attack logs. Were the thresholds correct?

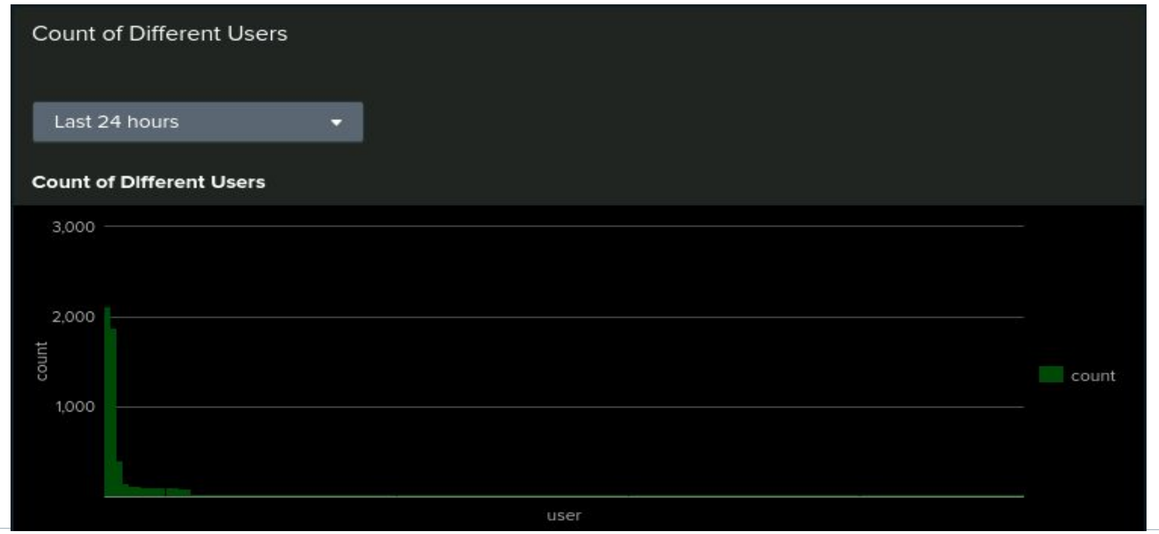
- Our findings showed that the threshold for the Hourly Failed Activities alert was accurate, as it successfully identified suspicious behavior.
- Although the other two alerts were not triggered, we believe the thresholds were appropriate. The attackers' activity blended into typical daily behavior, and lowering the thresholds further would have led to excessive alert fatigue.
- This emphasizes the importance of not relying solely on alerts for security. Instead, alerts should be one of several tools used to monitor and defend networks, systems, and environments effectively.

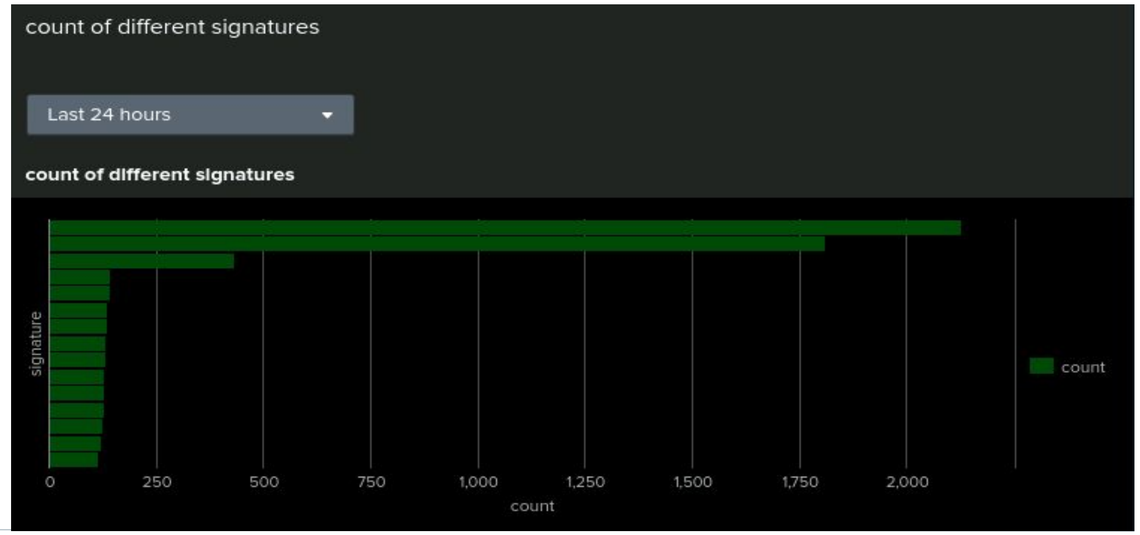
Attack Summary—Windows

Summarize your findings from your dashboards when analyzing the attack logs.

- While our alerts struggled to catch the attacker, our dashboards proved much more effective.
- By analyzing the attack logs visually, we could easily identify spikes in suspicious activity over two sustained three-hour intervals. The attackers spread out their activity to avoid triggering our alerts, but these patterns were clearly visible on the dashboards.
- We observed notable spikes from 12 AM to 3 AM and 8 AM to 11 AM, with increased user account lockouts, password reset attempts, and successful logins. This highlights the value of dashboards in identifying subtle attack patterns that alerts might miss.

Screenshots of Attack Logs

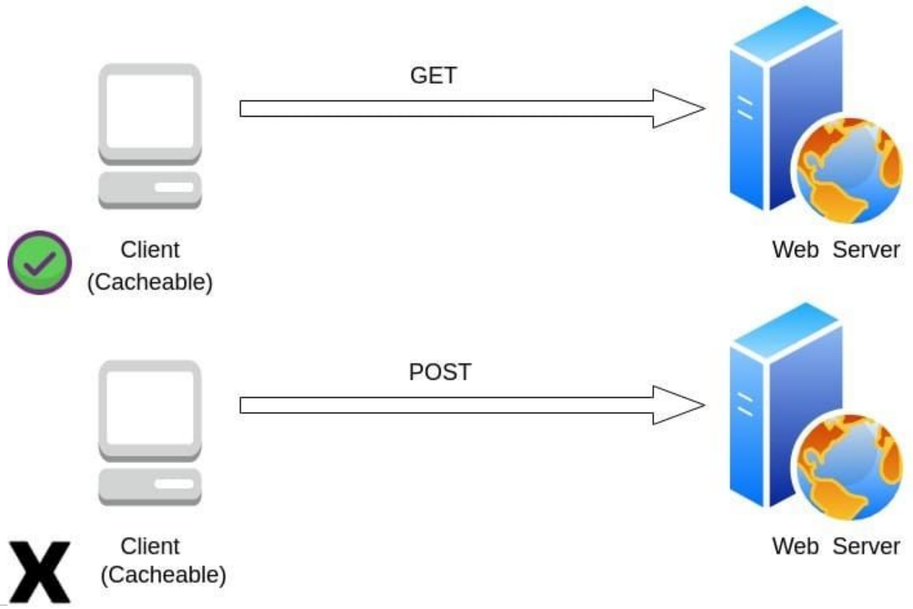




Attack Summary—Apache

Summarize your findings from your reports when analyzing the attack logs.

- The GET and POST methods had drastic changes in their average numbers over time. GET 9851 > 3151 and POST 106 > 1324
- There were two instances of high POST activity: 6PM (730) and 8PM (1415)
- There are large spikes of GET requests and then one large POST method at 8pm (1296)
- There was a significant drop in referrer domains



Attack Summary—Apache

Summarize your findings from your alerts when analyzing the attack logs. Were the thresholds correct?

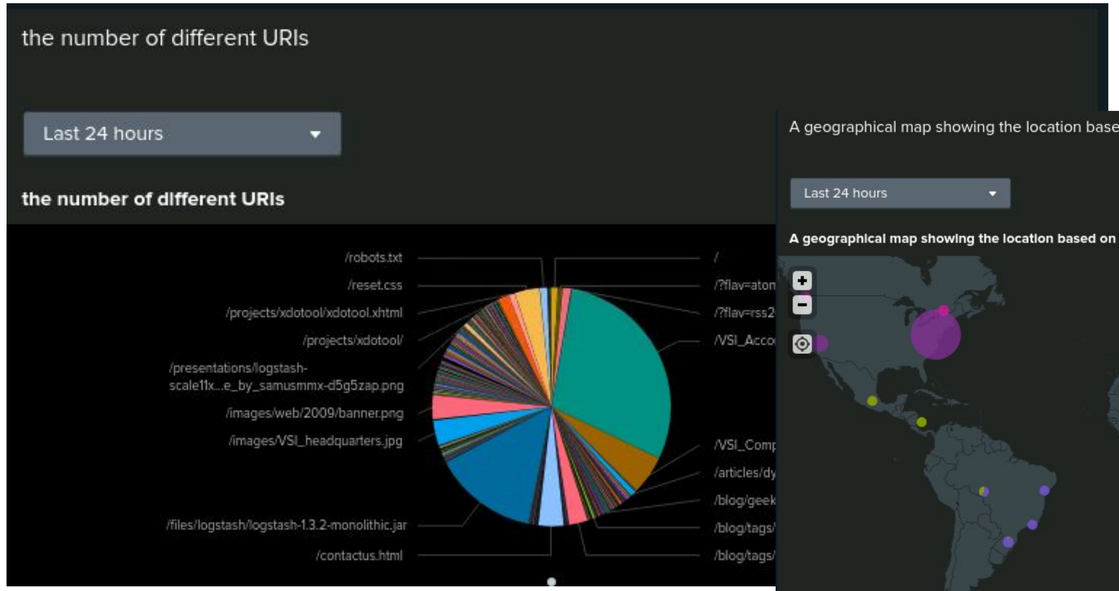
- Significant drop in all response codes (primarily 200 codes) - 404 codes drastically increased
- Around 10pm on March 25, 2020 there were roughly 937 events from international sources compared to the rest of the day which were 120 or less
- There was a huge increase in locational access from Kharkiv (432) and Kiev (439), Ukraine
- There were two large chunks of URI data that appeared on the chart for /VSI_Account_logon.php and /files/logstash/logstash-1.3.2-monolithic.jar

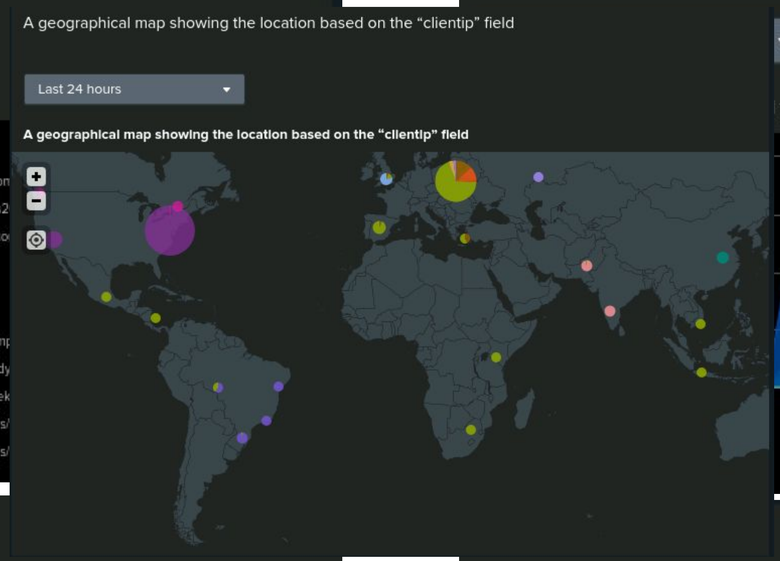
Attack Summary—Apache

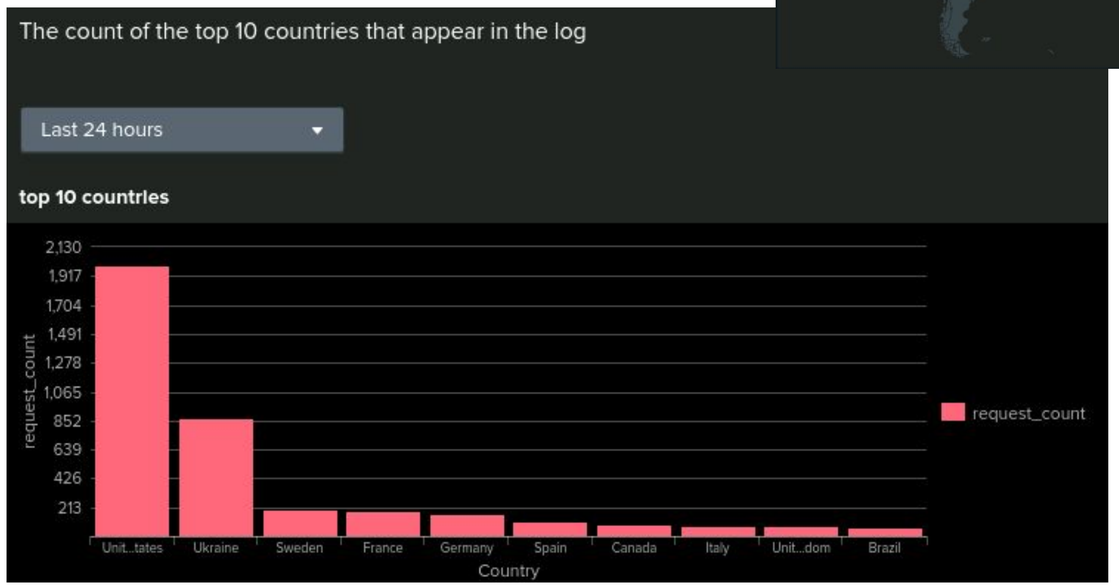
Summarize your findings from your dashboards when analyzing the attack logs.

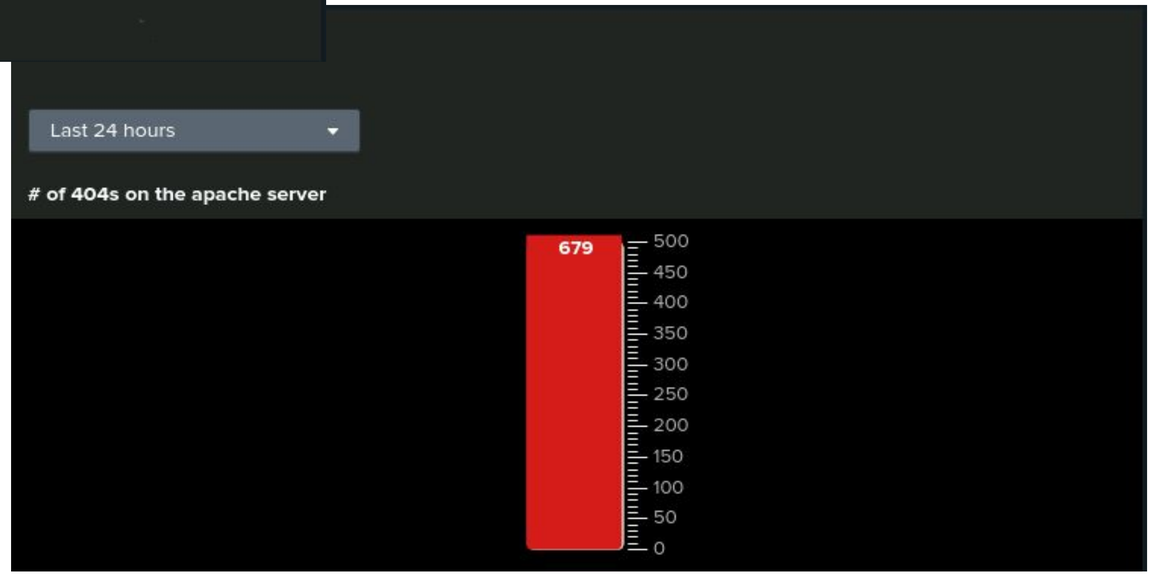
- In this instance, what seems like a Bruteforce attack has occured. This is based on the number of failed logins, POST methods and vector of /VSI_Account_logon.php

Screenshots of Attack Logs









Summary and Future Mitigations

Project 3 Summary

- What were your overall findings from the attack that took place?

In general, baseline reports should not be expected to be a “catch-all” for incidents and it is imperative to have monitoring with digestible visuals for more intimate views of the data as well as a live stream of data in order to catch the vectors while they are in motion as opposed to days later.

- To protect VSI from future attacks, what future mitigations would you recommend?

1. Enable automatic account lockouts after a set number of failed password attempts.
2. Enforce more strict password rules for stronger security.
3. Restrict admin permissions to only what's necessary.
4. Limit connections from unexpected countries or regions.
5. Restrict repeated connection attempts from a single source within a defined timeframe.