Splunk: Exploring Splunk

Introduction

Splunk is a powerful Security Information and Event Management (SIEM) solution widely used for searching, monitoring, and analyzing machine-generated big data via a web-style interface. It helps cybersecurity professionals extract actionable insights from large volumes of logs and events. At the core of Splunk is Search Processing Language (SPL), a specialized query language designed to provide precise, real-time insights from complex datasets. SPL enables users to form powerful search queries by using various functions and commands, allowing both simple and advanced analyses.

This room explores the fundamentals of using SPL to craft effective searches. It covers applying filters to narrow down results, using transformational commands to manipulate data, and altering the order of results to obtain the most relevant insights. By chaining SPL queries, users can address both simple and complex search tasks efficiently, making it an essential tool for anyone working with large datasets in cybersecurity.

Objective

The objective of this room was to enhance practical knowledge of Splunk's Search Processing Language (SPL) and its application in analyzing machine data. This room focused on using the **Search & Reporting App** to explore and refine data efficiently. Key topics included learning how to filter results effectively using SPL, structuring search outputs to provide clarity and relevance, and leveraging transformational commands to manipulate data and gain deeper insights. By mastering these fundamentals, participants could construct both simple and complex queries, making SPL an indispensable tool for log analysis and cybersecurity investigations.

First Task: Identifying the Host

Objective

The objective of this task was to utilize Splunk's **Data Summary** tab to identify the host responsible for generating the log data.

Approach

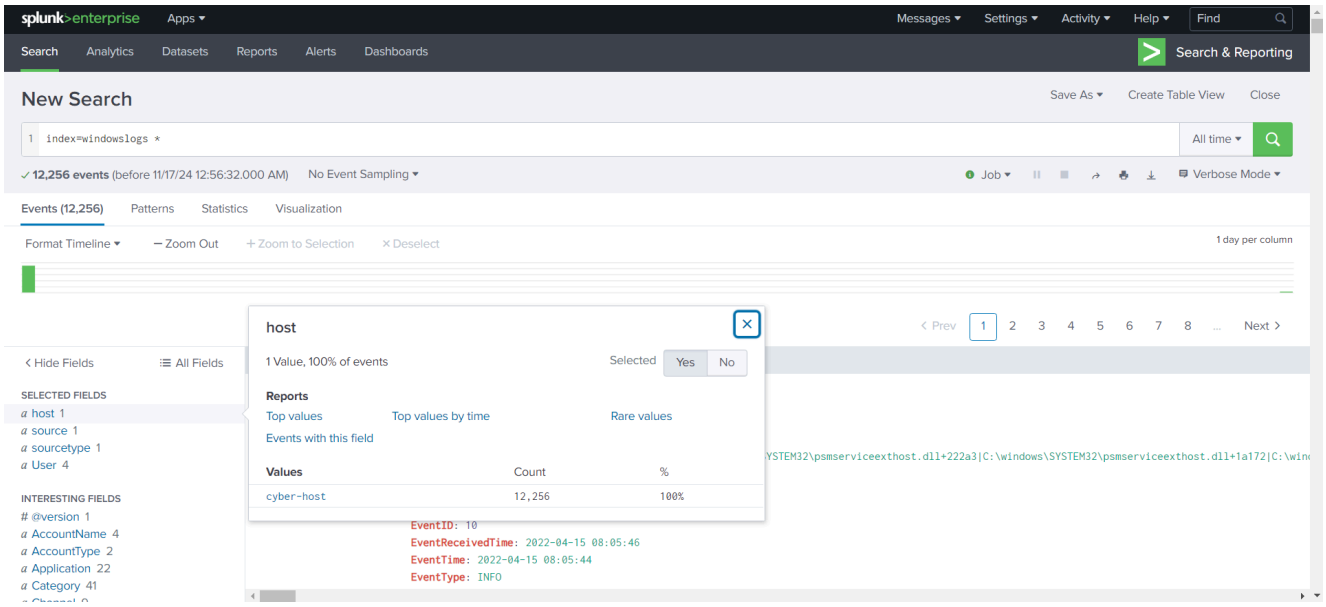
The task began with navigating to the **Data Summary** tab in Splunk, which provides a high-level overview of datasets, including hosts, sources, and sourcetypes. A query was executed using the following SPL command:

“index=windowslogs *”

This command retrieved all events within the `windowslogs` index, enabling a comprehensive analysis of the available data.

Result

The query results revealed that the host generating the logs was cyber-host.



Task 2: Filtering and Analyzing Search History

Objective

The goal of this task was to explore and filter search history to extract specific details, such as the 7th search query, the IP address with the highest recorded events, and the number of events within a specified timeframe.

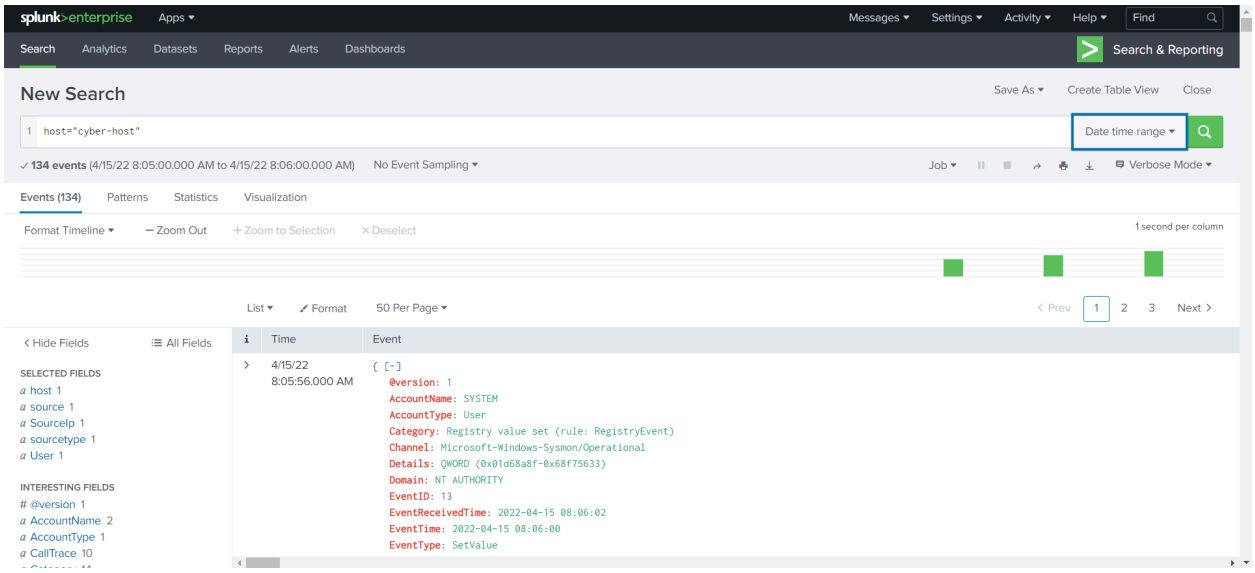
Approach

1. **Finding the 7th Search Query:**
   The search history was filtered using the "All-time" timeframe to display all past search queries. From the displayed list, the 7th query was identified for further analysis.
2. **Identifying the IP Address with Maximum Events:**
   The **sourceIP** field was enabled in the left-hand field box to refine the results. By analyzing the data, the IP address with the maximum number of recorded events was identified.
3. **Filtering Events by a Specific Timeframe:**
   The search was further refined to a specific timeframe by filtering events within the date **04/15/2022** and time range **08:05 AM to 08:06 AM**. This helped isolate and count the number of events occurring during that period.

Results

- The 7th search query in the "All-time" search history was successfully located.
- The IP address with the maximum recorded events was identified from the dataset.
- A specific number of events were found within the timeframe of **04/15/2022, 08:05 AM to 08:06 AM**.

Screenshot:



Task 3: Filtering Events and Analyzing Data

Objective

The objective of this task was to perform advanced filtering of log events using specific fields and search parameters. This included analyzing event counts by criteria like EventID, AccountName, DestinationIP, DestinationPort, and leveraging SPL's wildcard functionality.

Approach

1. **Filtering by EventID and AccountName:**
   Events were filtered based on their EventID and the `AccountName` field. This step provided insight into user-specific activities and event types.
2. **Counting Events by Destination IP and Port:**
   A query was executed to find the total number of events matching a specific `DestinationIP` and `DestinationPort`. This helped pinpoint targeted network activities within the dataset.

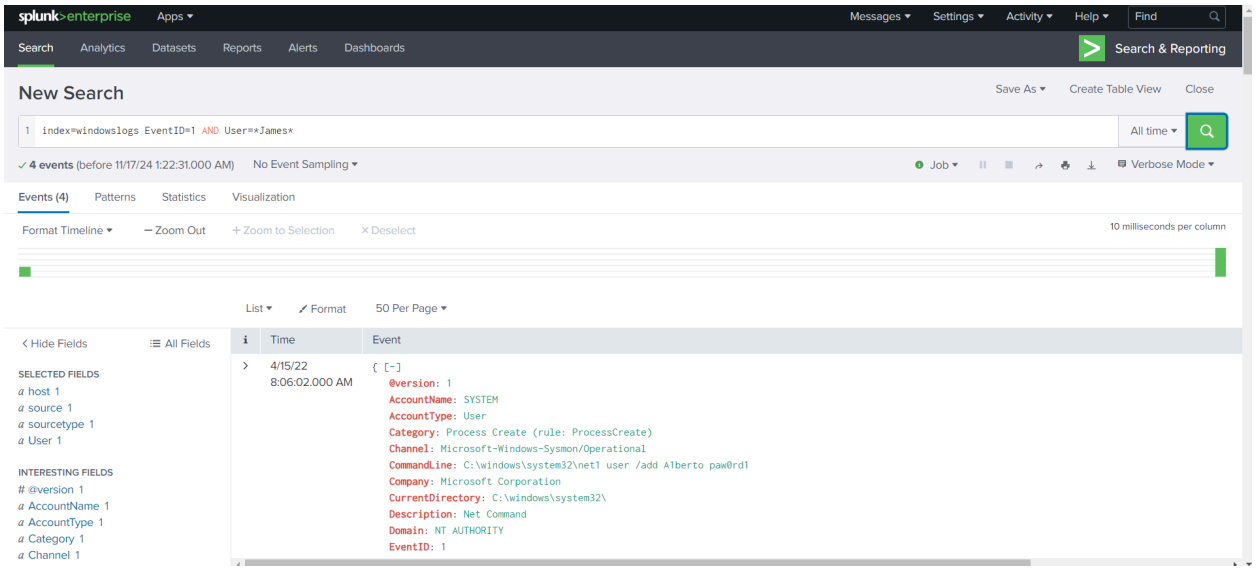
3. **Finding the Source IP with the Highest Count:**
   Using the following SPL query: index=windowslogs Hostname="Salena.Adam" DestinationIp="172.18.38.5"
4. The source IP with the highest event count was identified. This step provided valuable data on the origin of significant traffic associated with the specified host and destination.
5. **Counting Events Containing the String "cyber":**
   A search was conducted to identify the total number of events that included the keyword "cyber," helping focus on cybersecurity-related logs.
6. *Counting Events Using the Wildcard "cyber":**
   By applying the wildcard feature (`cyber*`), the search expanded to include variations like "cybersecurity" and "cyberattack," providing a broader analysis of related events.

Results

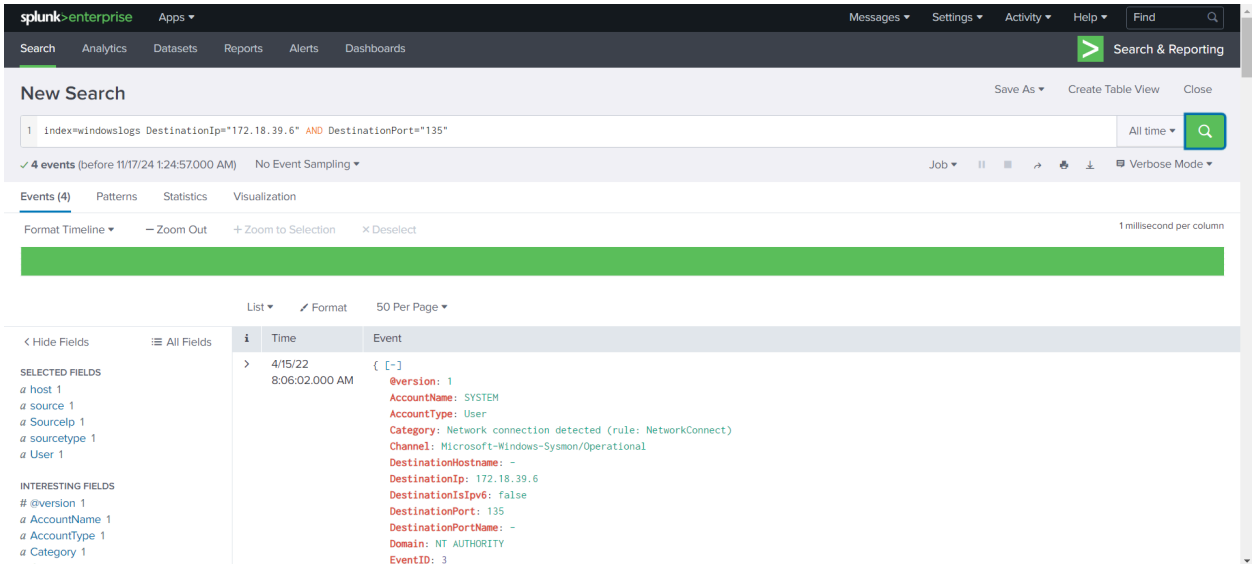
- Events filtered by **EventID** and **AccountName** were successfully analyzed.
- The count of events with the specific `DestinationIP` and `DestinationPort` was determined.
- The source IP with the highest event count for the specified query was identified.
- A total count of events containing the string "cyber" was obtained.
- Events containing variations of the string "cyber*" using the wildcard feature were successfully identified.

Screenshot(s):

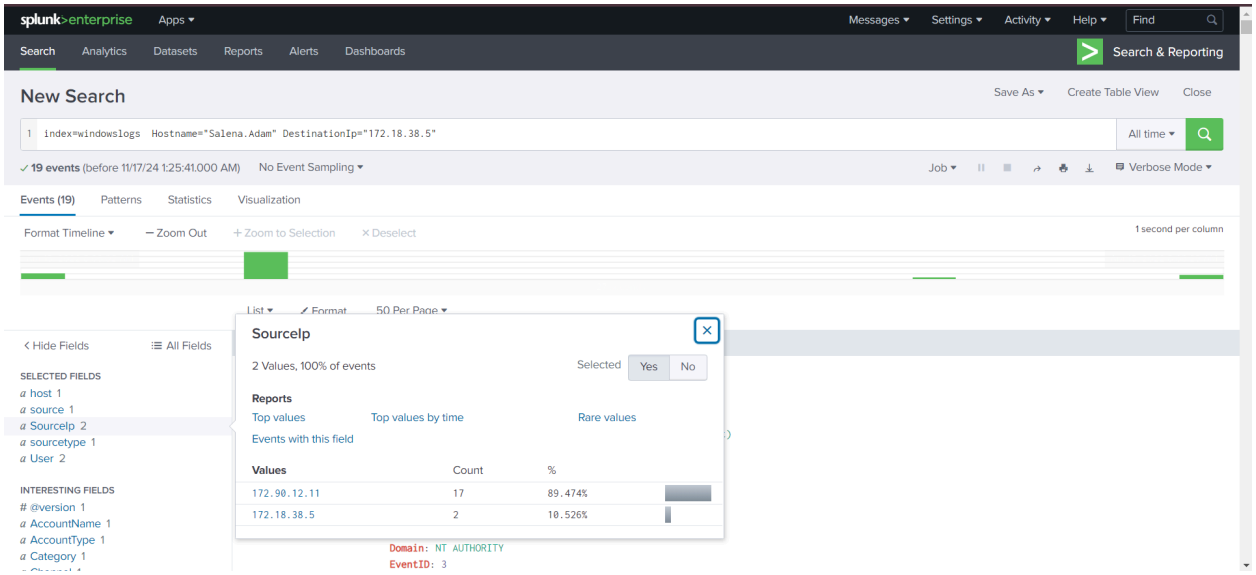
EventID = 1 AND User = James



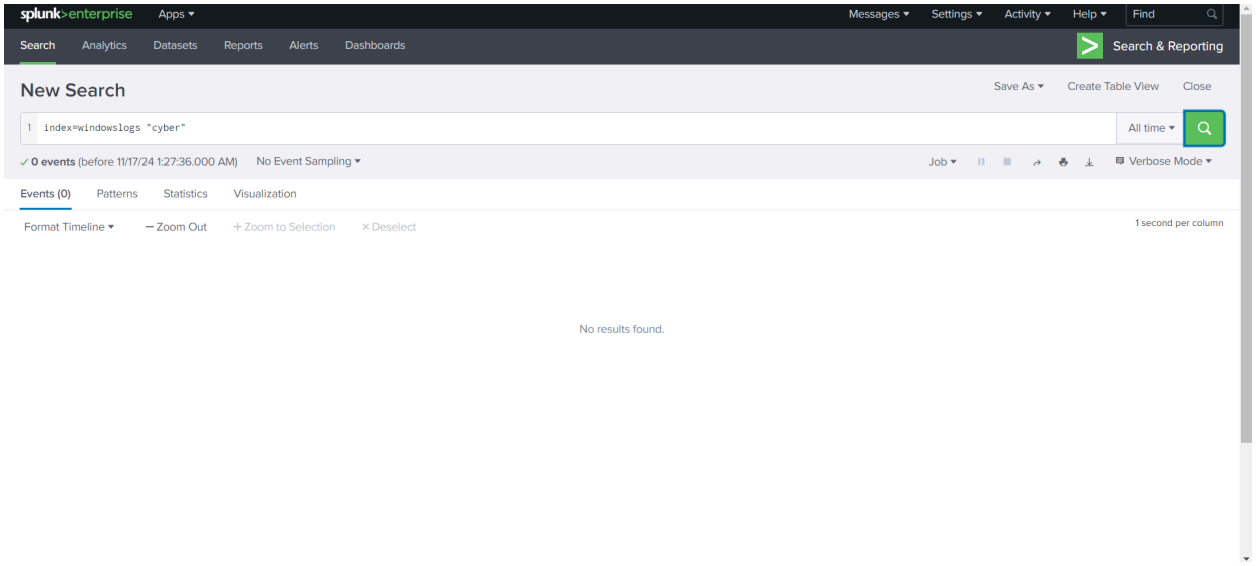
Destination IP 172.18.39.6 AND destination Port 135



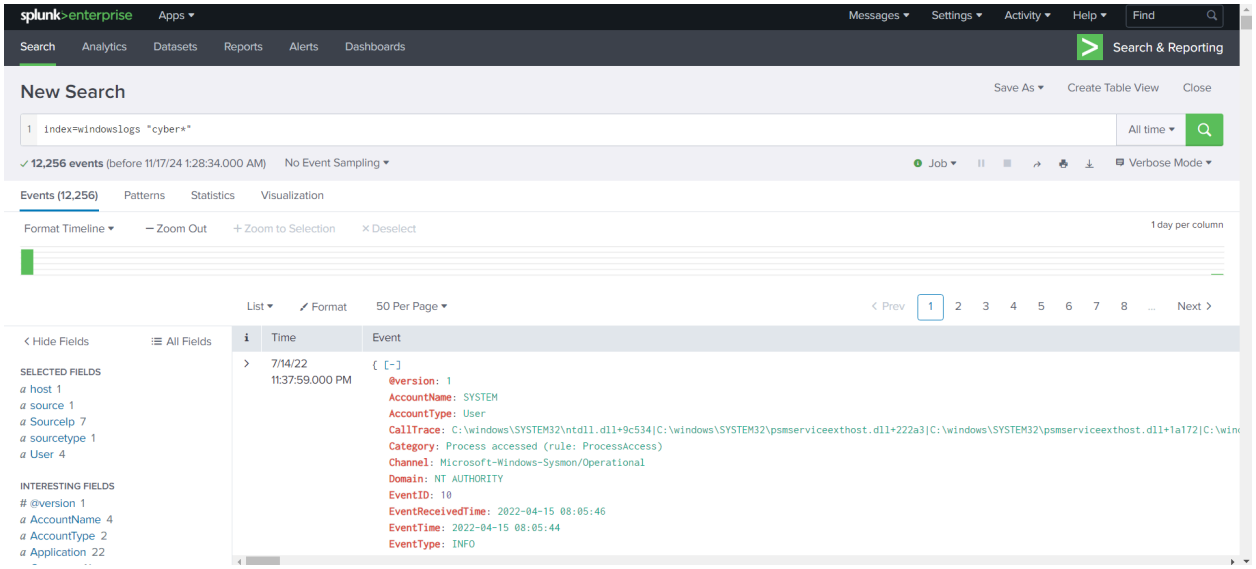
index=windowslogs Hostname="Salena.Adam" DestinationIp="172.18.38.5"



Term “cyber”



Cyber Wildcard:



Task 4: Query Optimization with Piping, Reverse, and Dedup

Objective

This task focused on enhancing data organization by piping multiple filters into a query. The goal was to display event details using the `reverse` keyword to reorder results and the `dedup` keyword to eliminate duplicate entries.

Approach

Constructing the Query with Multiple Filters:

A piped query was executed to retrieve specific fields such as `_time`, `EventID`, `Hostname`, and `SourceName` from the `windowslogs` index. The query used was:

```
index=windowslogs | table _time EventID Hostname SourceName | reverse
```

- The `table` command was used to limit the displayed fields for a cleaner output.
- The `reverse` keyword reordered the results in descending order based on the `_time` field.

Applying the Dedup Keyword:
The same query was extended with the `dedup` keyword to remove duplicate rows. This ensured each unique event was displayed only once. The updated query was:

```
index=windowslogs | table _time EventID Hostname SourceName | reverse
| dedup _time
```

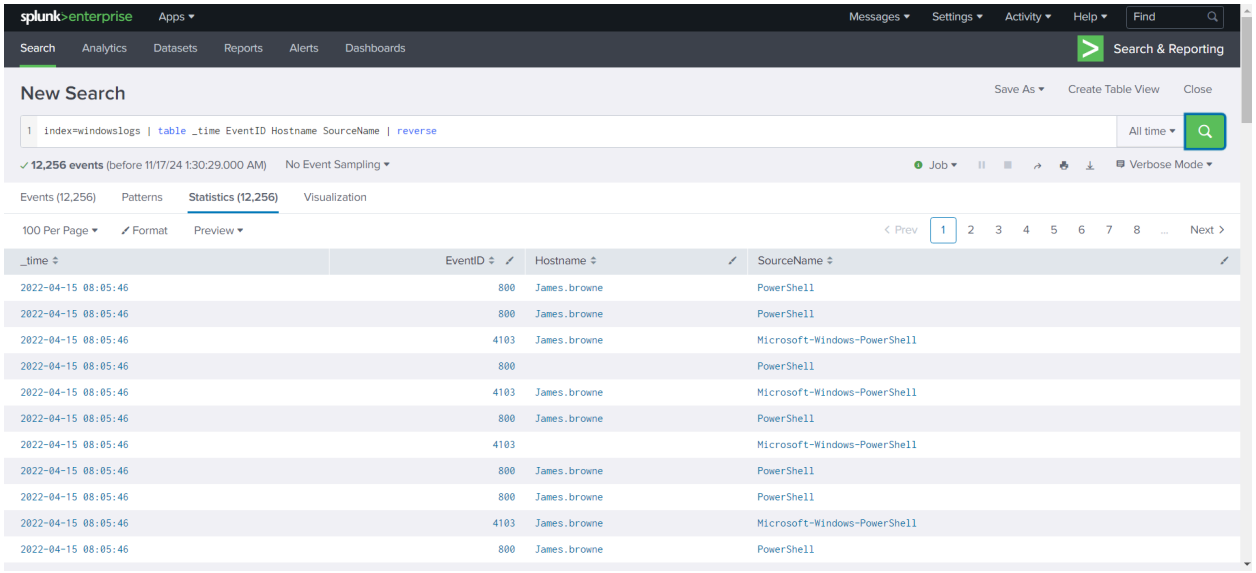
- The deduplication was applied to the `_time` field, ensuring unique timestamps were retained.

Results

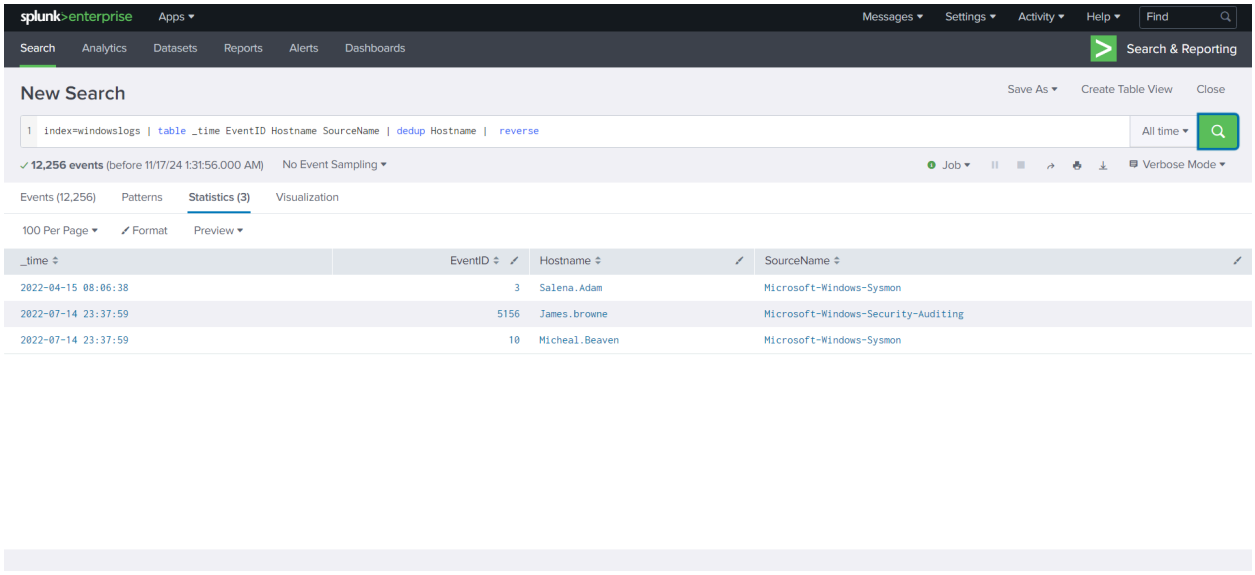
- The piped query successfully filtered and structured the event data.
- The `reverse` keyword reordered the results by reversing the chronological order.
- The `dedup` keyword removed duplicate entries, leaving only unique event records.

Screenshot(s):

Using the reverse keyword:



Using the dedup keyword:



Task 5: Query Enhancements with Reverse, Tail, and Sort Commands

Objective

This task aimed to analyze log data by utilizing the `reverse`, `tail`, and `sort` commands to identify specific HostNames, EventIDs, and SourceNames from the dataset.

Approach

Using the Reverse Command to Identify the Top HostName:
The following query was executed to reverse the chronological order of events and display specific fields:

```
index=windowslogs | table _time EventID Hostname SourceName | reverse
```

1. By applying the `reverse` command, the HostName appearing at the top of the results was identified.

Finding the Last EventID with the Tail Command:
The query was updated to include the `tail` command, which returns the last event(s) from the reversed result set. The query used was:

```
index=windowslogs | table _time EventID Hostname SourceName | reverse
| tail 1
```

2. This provided the last EventID from the reversed list.

Sorting the Results by SourceName:
To determine the top SourceName, the query was further refined using the `sort` command:

```
index=windowslogs | table _time EventID Hostname SourceName | reverse
| sort SourceName
```

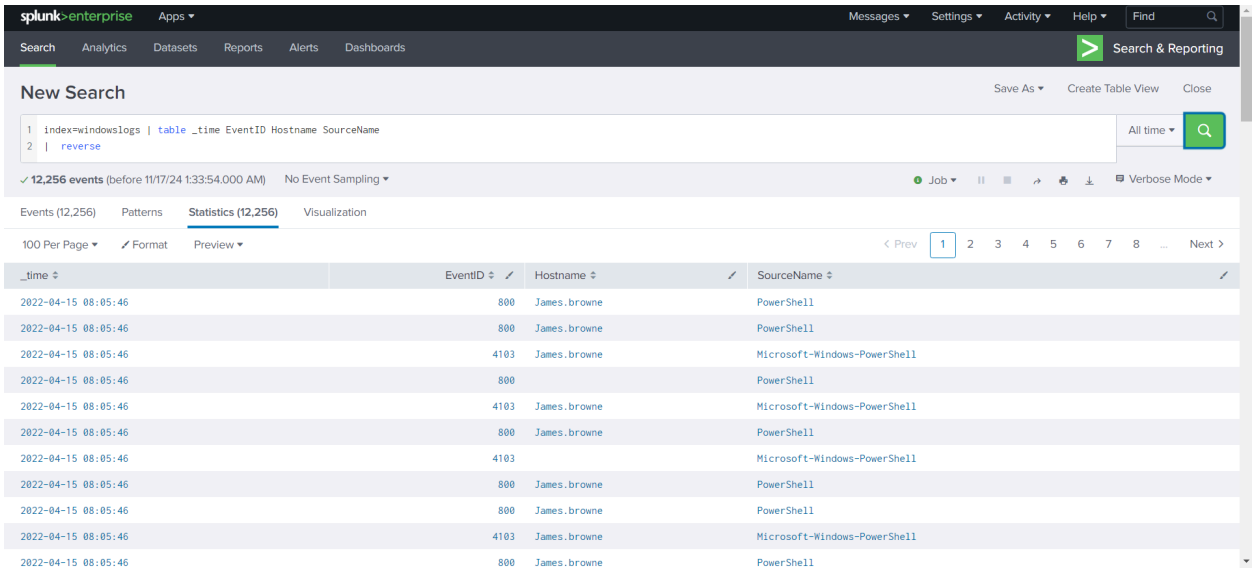
3. The sorted results revealed the top SourceName in alphabetical order.

Results

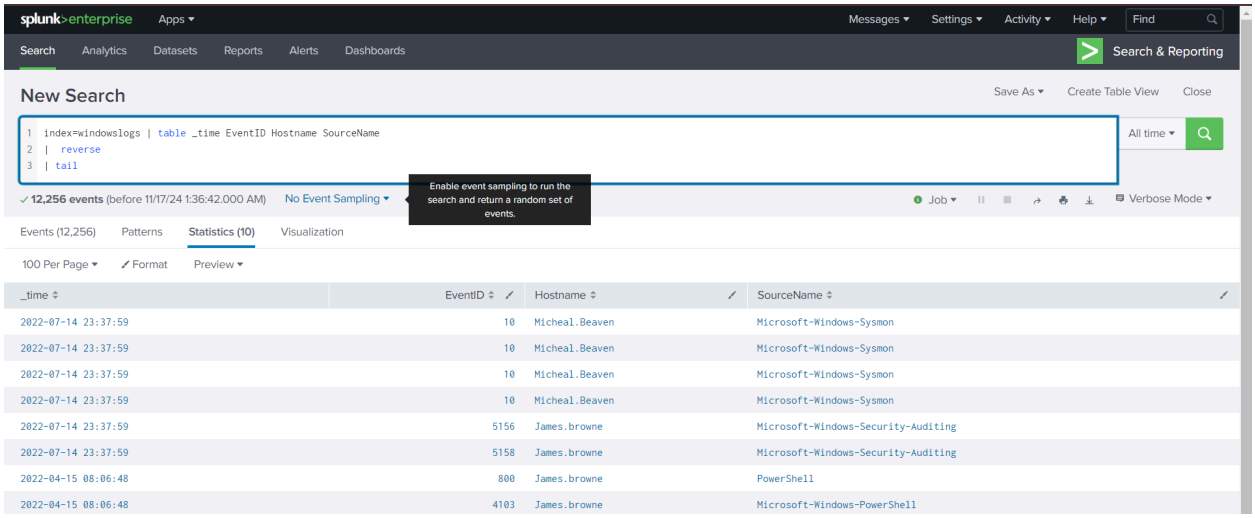
- The **HostName** appearing at the top of the reversed list was identified.
- The **last EventID** returned from the `tail` query was determined.
- The **top SourceName** from the sorted results was highlighted.

Screenshot(s):

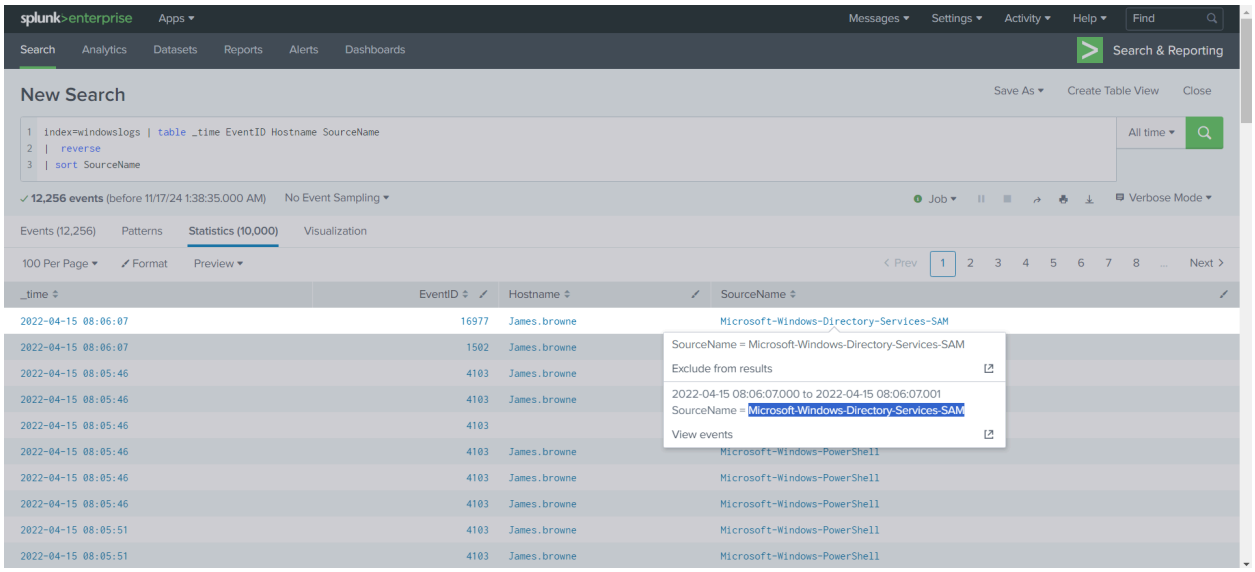
Adding Reverse to the query:



Appending tail to the above query:



Sorting the results using sort:



Task 6: Advanced Analysis with Top, Rare, and Chart Commands

Objective

This task focused on utilizing advanced SPL commands—`top`, `rare`, and `chart`—to analyze image processes, user activity, and visualize data in Splunk.

Approach

Listing the Top 8 Image Processes Using the Top Command:
The `top` command was employed to list the top 8 image processes and their respective counts:
`index=windowslogs | top limit=8 Image`

1. From the results, the **6th Image process** was identified, and its total count recorded.

   Identifying the User with the Least Activities Using the Rare Command:
   To determine the user with the fewest recorded activities, the `rare` command was used:
   `index=windowslogs | rare limit=1 User`

2. This command provided the least frequently appearing user, highlighting minimal activity logs.

   Creating a Pie-Chart with the Chart Command:
   A pie chart was generated to display the count of events associated with each image process. The specific query to analyze the `conhost.exe` process was:
   `index=windowslogs | chart count by Image`

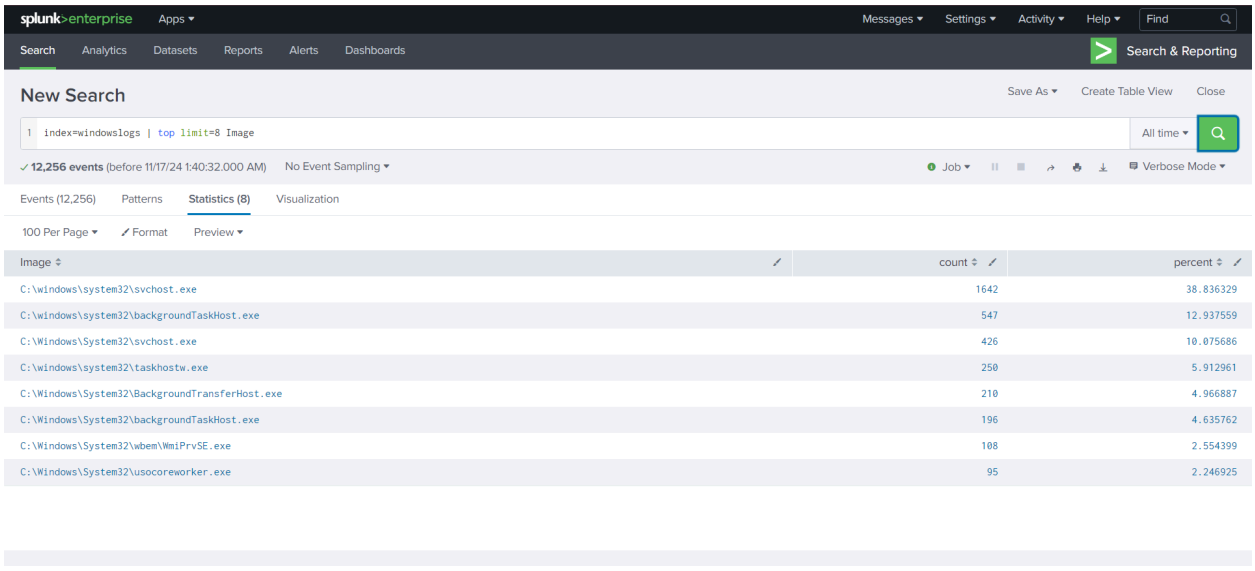
3. From the resulting chart, the count of events for `conhost.exe` was extracted.

Results

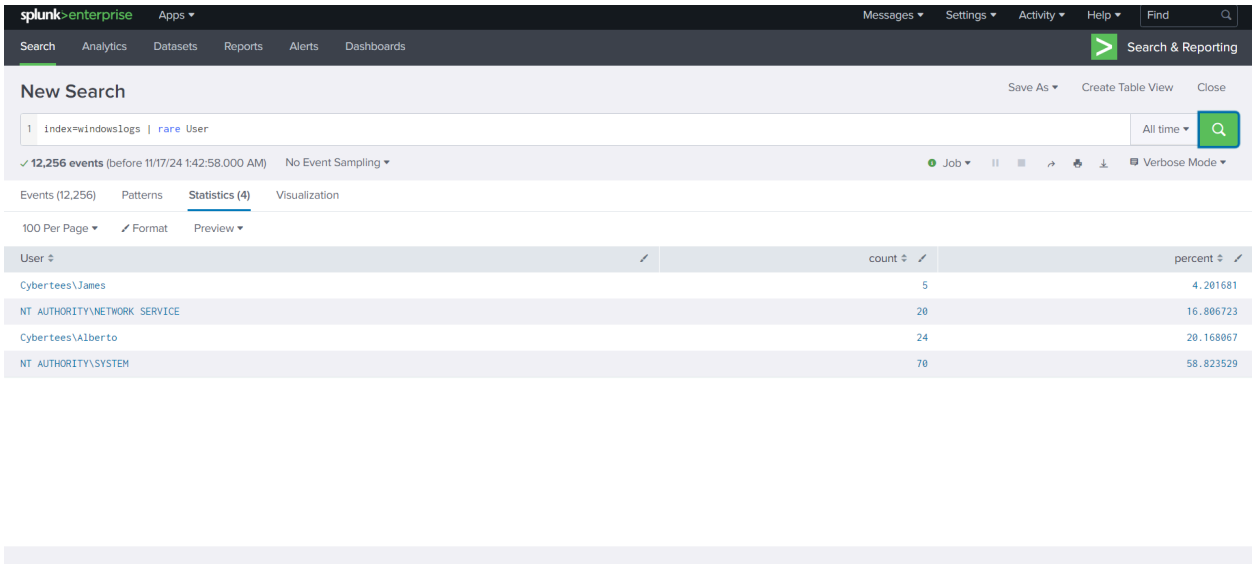
- **Top 8 Image Processes:** The 6th image process and its count were identified using the `top` command.
- **Least Active User:** The user with the least activity was found using the `rare` command.
- **Count for Conhost.exe:** The total count for the `conhost.exe` process was determined from the pie-chart results.

Screenshot(s):

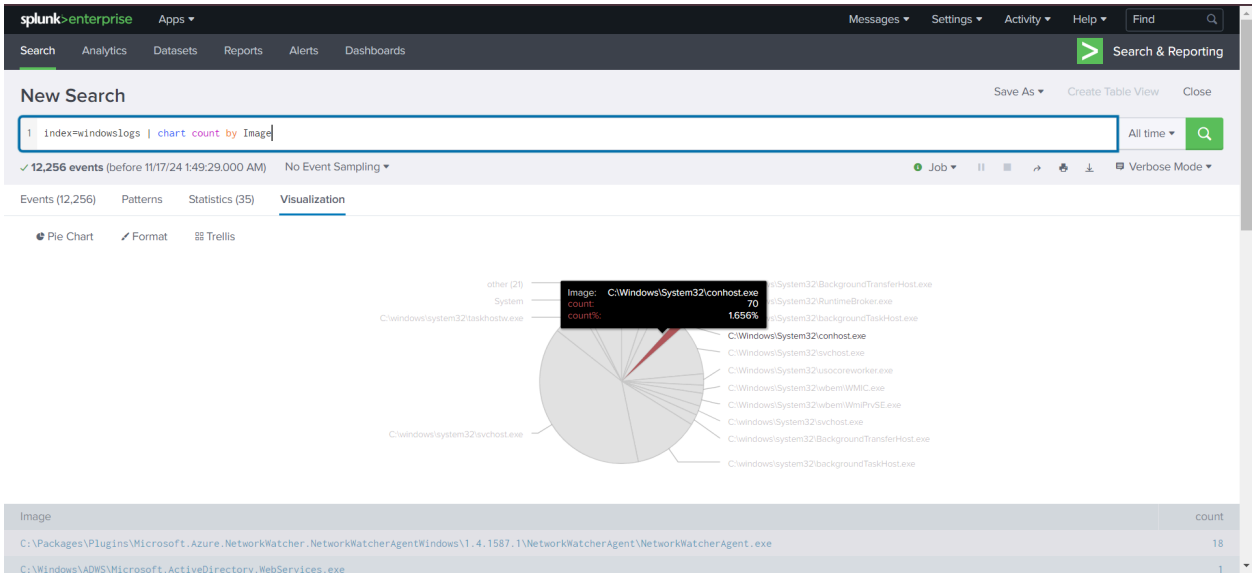
Top 8 of the search:



Using the rare keyword on user:



Visualizing the data into a chart based on the specific processes:



Conclusion

This wraps up the Splunk SPL room. Throughout the tasks, we explored the core functionality of Splunk's Search Processing Language (SPL) and how it can be leveraged to search, filter, and analyze large datasets efficiently.

This room covered various SPL commands and functions, such as `reverse`, `dedup`, `top`, `rare`, and `chart`, to refine search results, identify patterns, and visualize data. By chaining these commands together, we were able to tackle tasks ranging from identifying the most frequent image processes to determining the least active users, and even creating insightful visualizations.

Mastering these SPL commands and techniques is essential for navigating and analyzing machine data within Splunk. With these skills, one can enhance their ability to detect anomalies, monitor system activities, and respond to security events more effectively.