Lucas McMahon
6/27/25

FixIT Challenge – Splunk Lab Write-Up

Table of Contents

Introduction

In this lab, I took on the role of John, a candidate applying for a SOC-L2 position at MSSP Cybertees Ltd. My final challenge was to resolve event parsing and data extraction issues within a Splunk instance. This test was designed to simulate real-world log ingestion and analysis problems and evaluate my ability to diagnose and fix configuration issues.

The hands-on experience focused on identifying improperly formatted events, adjusting Splunk configuration files, writing effective regular expressions, and extracting meaningful fields for analysis.

Objectives

- Understand and resolve event boundary issues in Splunk
- Configure props.conf and associated files to handle multi-line events
- Write and test regex patterns for field extraction
- Extract critical fields: Username, Department, Domain, Source_IP, Country
- Perform basic event analysis and answer targeted questions

- Gain familiarity with troubleshooting log ingestion problems in Splunk

Lab Environment.

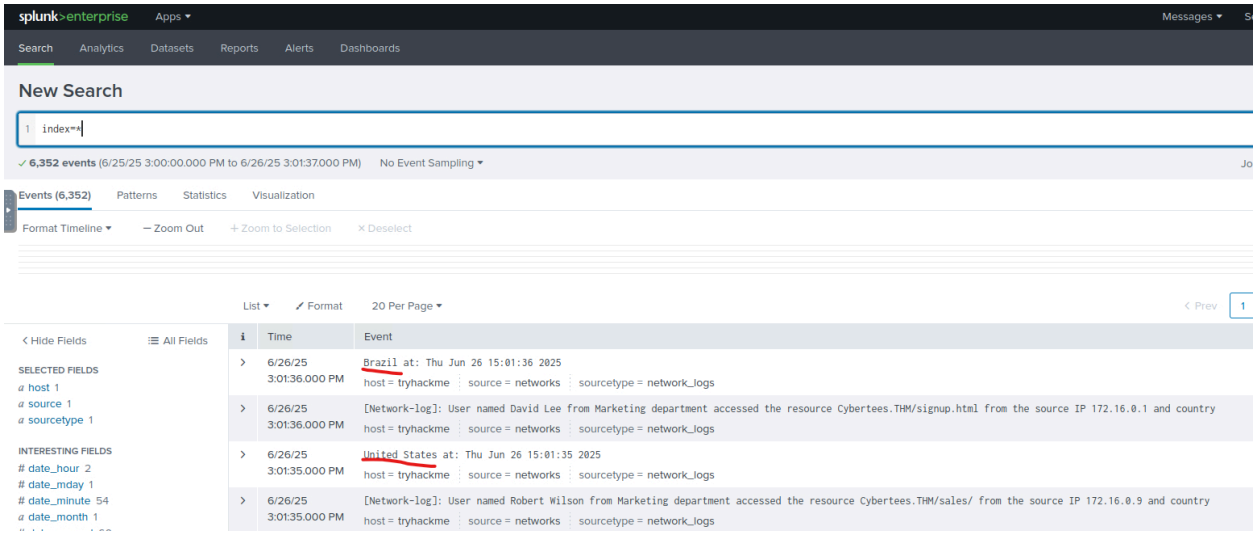
- Virtual machine hosted by TryHackMe (accessible via split-screen view)
- Splunk Web Interface: `http://MACHINE_IP:8000`
- Splunk App: `Fixit`
- Splunk installation path: `/opt/splunk`
- Data source: Custom script outputting multi-line logs

Tools and Technologies Used

- **Splunk Enterprise** (Log ingestion, configuration tuning, field extraction)
- **props.conf, transforms.conf, fields.conf** (for parsing and transformation)
- **Regex** (Pattern-based field extraction)
- **Linux Terminal** (Navigating and editing configuration files)
- **ChatGPT** (Assistance with regex syntax and config structure)

Lucas McMahon
6/27/25

Step-by-Step Process



Taking a first look at our logs, we see that the events are improperly formatted. Two logs are jammed into one, and the event boundaries are not defined correctly. For example, some logs begin with country information when they should start with `[Network-log]`.

In order to properly research and analyze events, the logs need to be structured into single, complete records. To do this, we need to edit the `props.conf` file and define the correct event breaking rules.

Initial Diagnosis:

The logs are coming in as multi-line events, but Splunk is unable to determine the start of a new event.

Step 1: Review Default props.conf Suggestions

Using ChatGPT, I reviewed how to format the `props.conf` file.

Lucas McMahon
6/27/25

i am creating a splunk project, i have been given some sample logs to investigate and my first task is to fix the logs so that a multi event is combined into one event, i would like you to guide me through the steps of fixing this multi event by editing the props.conf file, here are sample logs: 6/20/25

✅ Step-by-Step Guide

1. Identify the sourcetype

Your sample shows:

plaintext — Copy — Edit

```
sourcetype = network_logs
```

We'll use this in the `props.conf` configuration.

2. Determine an appropriate event boundary

From your logs, each event seems to start with a timestamp in the format:

plaintext — Copy — Edit

```
6/20/25
3:06:36.000 PM
```

Splunk breaks events using **LINE_BREAKER** and groups lines together based on **SHOULD_LINEMERGE** or **BREAK_ONLY_BEFORE** directives.

3. Create/Edit `props.conf`

Here's the configuration to group each pair of lines into one event:

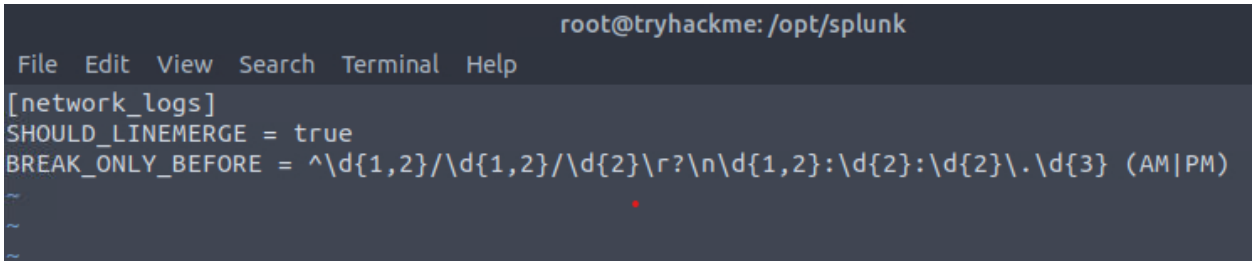
ini — Copy — Edit

```
[network_logs]
SHOULD_LINEMERGE = false
BREAK_ONLY_BEFORE = ^\d{1,2}/\d{1,2}/\d{2}\r?\n\d{1,2}:\d{2}:\d{2}\.\d{3} (AM|PM)
TIME_FORMAT = %m/%d/%y\n%I:%M:%S.%3N %p
MAX_EVENTS = 5
TRUNCATE = 10000
```

Here's what I learned about key parameters:

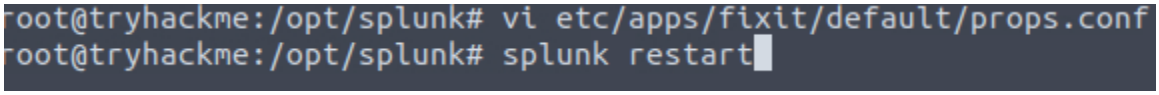
- `SHOULD_LINEMERGE = false` → Tells Splunk not to merge lines automatically
- `BREAK_ONLY_BEFORE` → Defines the regex pattern that indicates the start of a new event
- `TIME_FORMAT` → Helps Splunk extract timestamps correctly
- `MAX_EVENTS = 5` → Limits each event to 5 lines (adjustable)
- `TRUNCATE = 10000` → Prevents logs from being cut off if too long

Step 2: Create a Custom props.conf File
For this lab, I simplified the props.conf setup. I navigated to `/opt/splunk/etc/apps/fixit/default/` and created a new configuration for the source type.

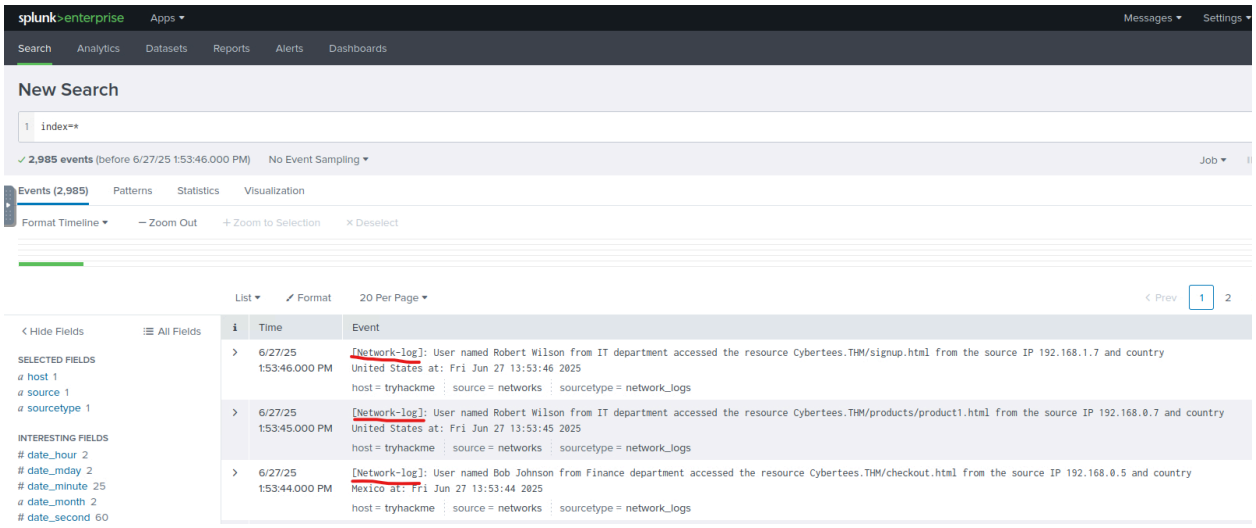


- I used `SHOULD_LINEMERGE = true` so Splunk could join lines unless instructed otherwise.
- The `BREAK_ONLY_BEFORE` stanza uses a custom regex pattern to mark the beginning of each log: `\[Network-log\]`.

Step 3: Restart Splunk
After saving the file, I restarted Splunk to apply the configuration changes.



As shown below, the logs now appear correctly grouped, each event starting with `[Network-log]` as expected.

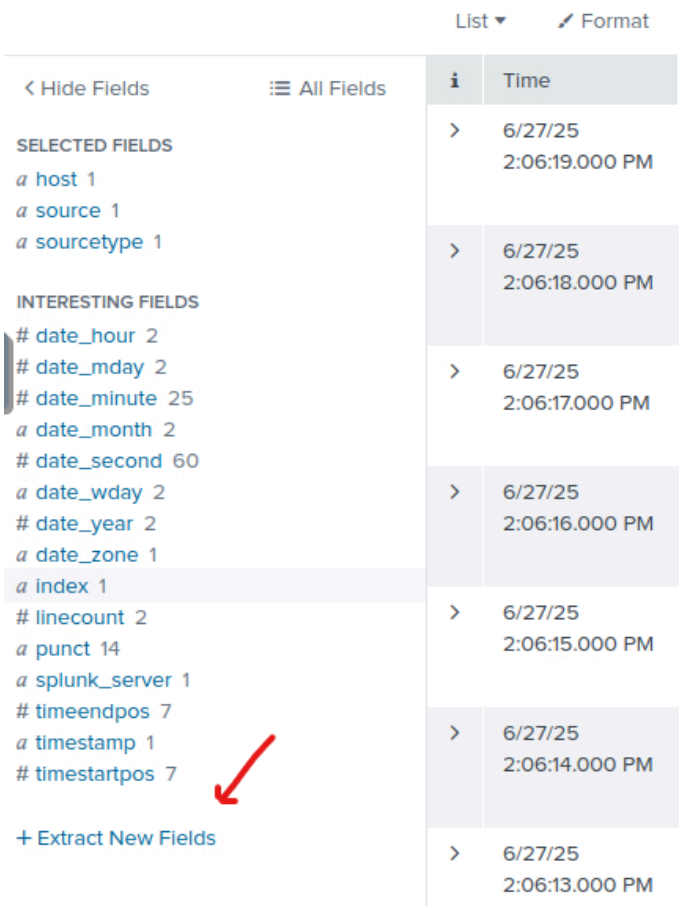


Step 4: Field Extraction via Regex
With events fixed, I moved on to field extraction to make analysis easier.

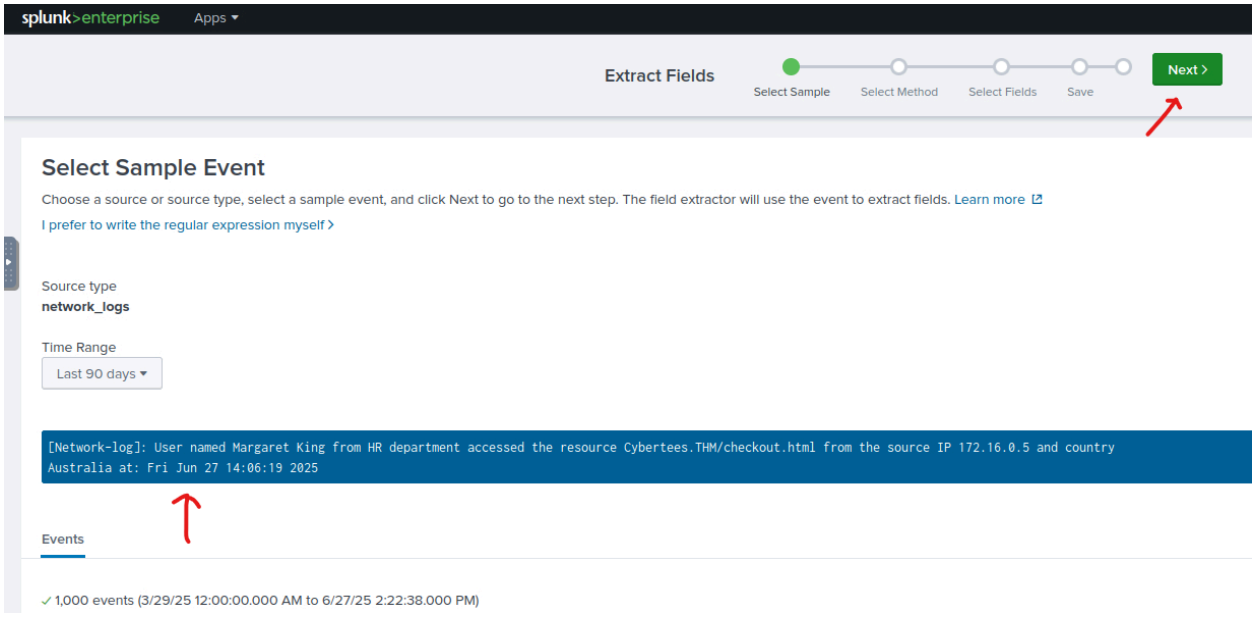
I launched the **Field Extractor** in Splunk and selected a sample log entry. I then used a regex pattern to pull the following fields:

- **Username**
- **Department**
- **Domain**
- **Source_IP**
- **Country**

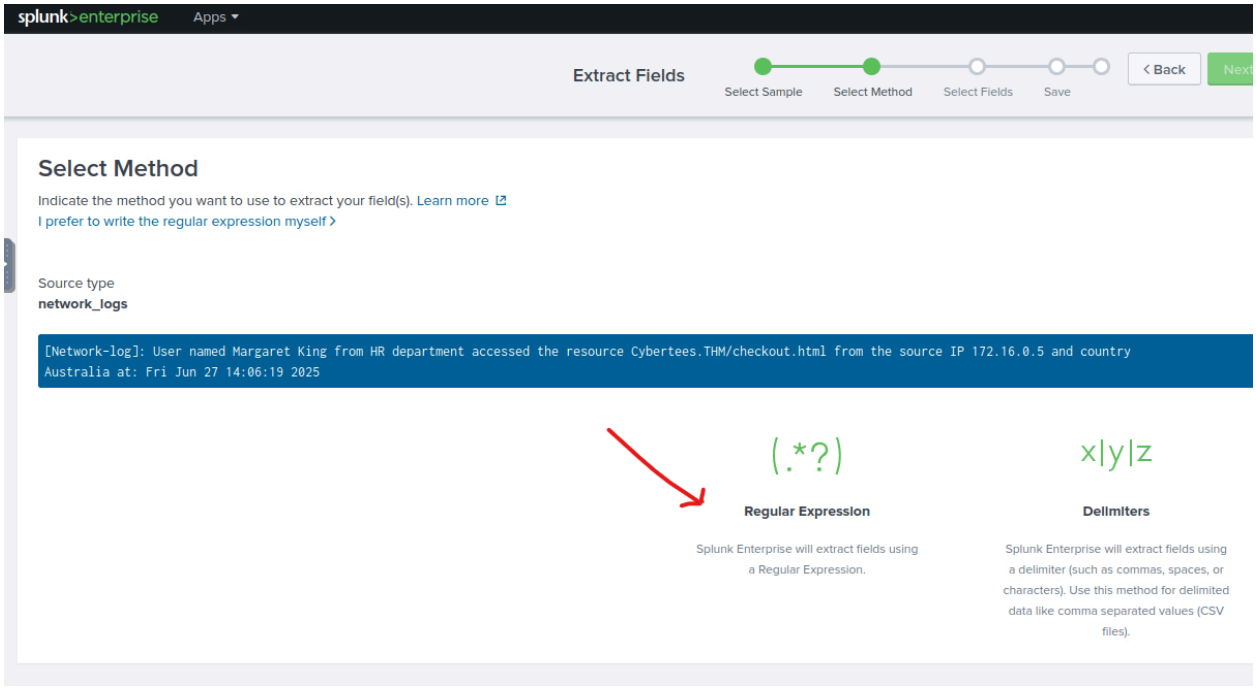
Lucas McMahon
6/27/25



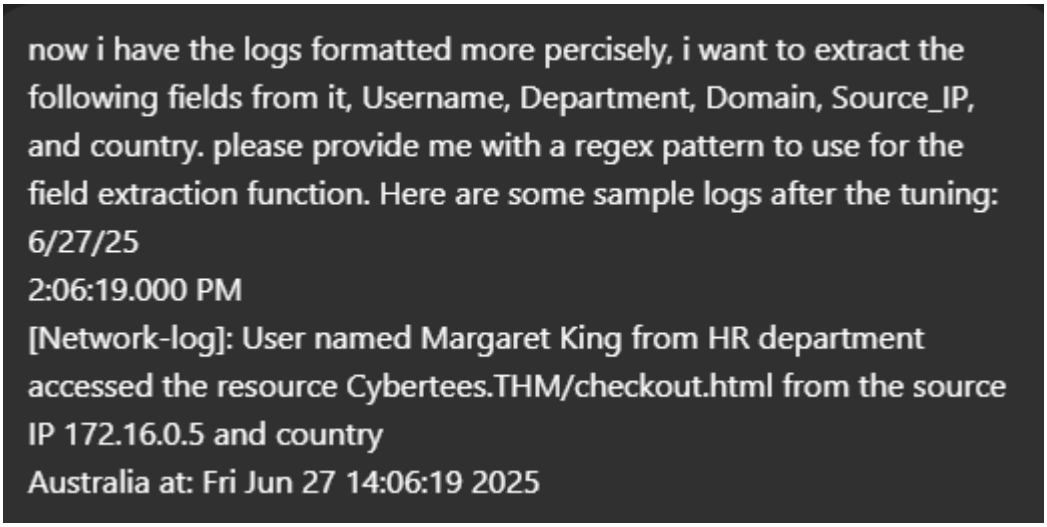
Starting by selecting a sample log, anyone will do, and clicking next.



Lucas McMahon
6/27/25



I created the regex with help from ChatGPT and tested it in Splunk.



Lucas McMahon
6/27/25

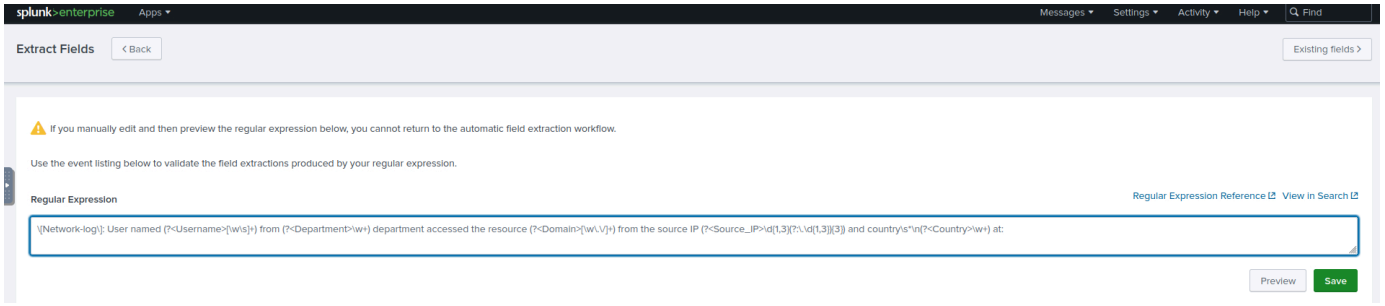
Thanks, Lucas — now that your logs are consistently formatted, here's the refined **regex pattern** you can use in Splunk's **Field Extraction** function to extract the five fields you mentioned.

✅ Regular Expression for Field Extraction

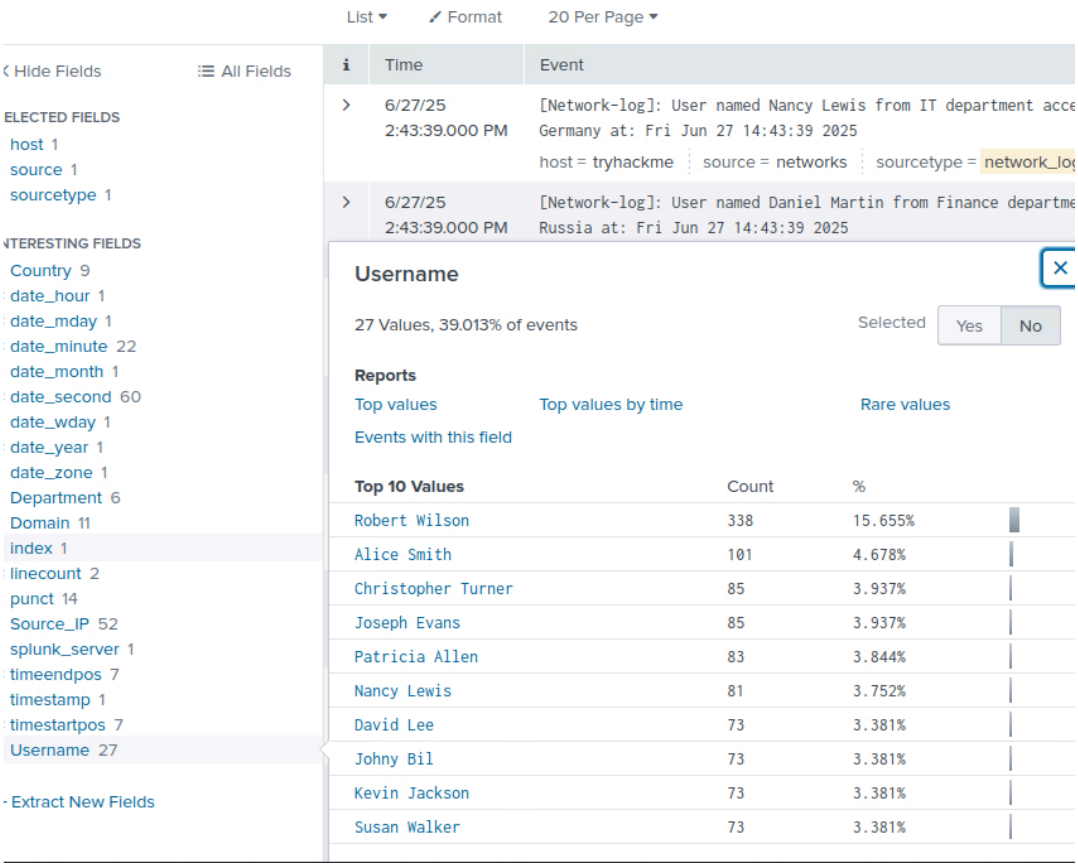
regex

Copy Edit

```
\[Network-log\]: User named (?<Username>[\w\s]+) from (?<Department>\w+) department accessed the r
```



The pattern successfully captured each field, allowing for quick searches and efficient event analysis.



Lucas McMahon
6/27/25

Results and Analysis

- Splunk now ingests and displays each log entry as a single, unified event.
- All five fields (Username, Department, Domain, Source_IP, Country) were accurately extracted using the custom regex.
- I was able to answer all the challenge questions using filtered search queries.

Example Insights:

- Number of unique usernames: **28**
- Top two countries user Robert accessed the domain from: **Canada, United States**
- File accessed by user: `secret-document.pdf`
- Number of unique departments: **6**
- Domain captured: **Cybertees.THM**

Challenges Faced

Identifying the correct event boundary for multi-line logs required experimentation with `BREAK_ONLY_BEFORE`.

Regex pattern construction was initially tricky due to the inconsistent formatting of logs, but ChatGPT helped fine-tune it.

Restarting Splunk to test changes was time-consuming and required patience for verification.

Maintaining accuracy while extracting fields, especially when fields like usernames and departments varied in length and content.

Conclusion

This lab taught me how to manipulate Splunk configuration files to fix log ingestion issues and apply regular expressions to extract meaningful fields from unstructured data.

Key takeaways include:

- Understanding and configuring `props.conf`, `fields.conf`, and `transforms.conf`
- Writing and testing regex patterns for log parsing
- The importance of structured data for effective log analysis in a SOC environment

This exercise strengthened my ability to troubleshoot real-world log issues and showcased practical Splunk skills that are critical in cybersecurity operations.

References

TryHackMe – FIXIT Challenge Room

Splunk Documentation: props.conf, Regex in Field Extraction

ChatGPT assistance with syntax guidance and regex optimization