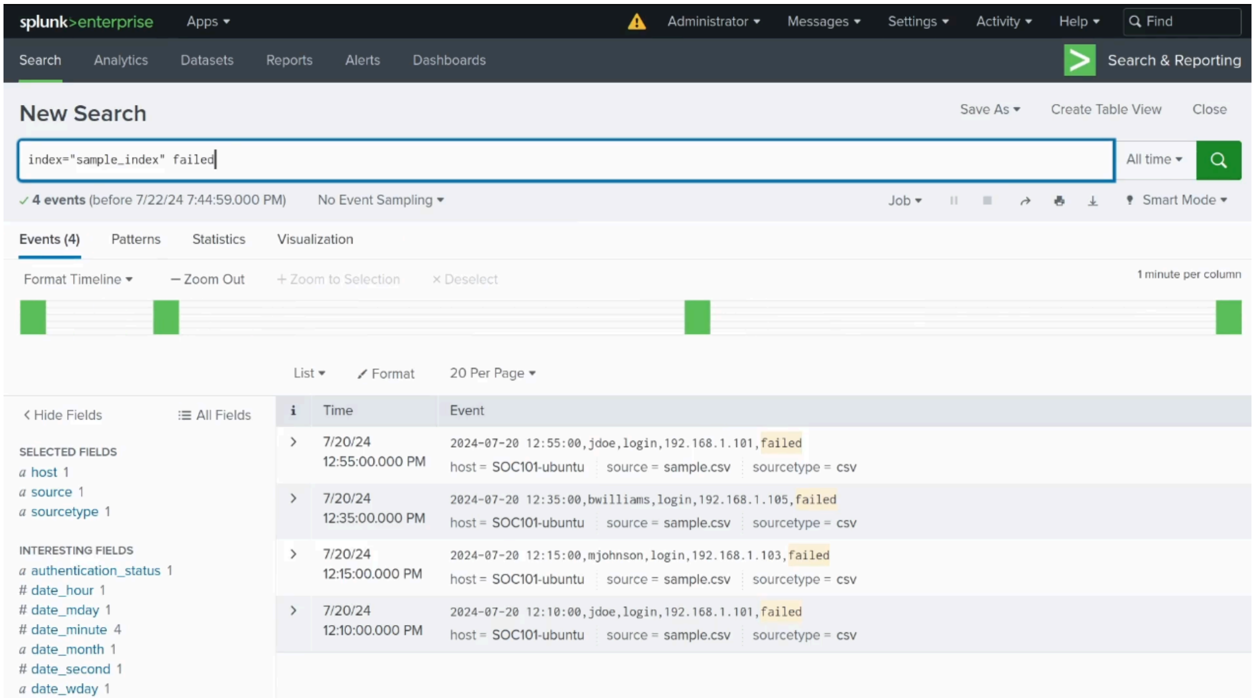
Splunk Important CMDs

1.) Splunk - Search Processing Language (SPL)

Splunk SPL searching UI:



```
index="sample_index" failed
```

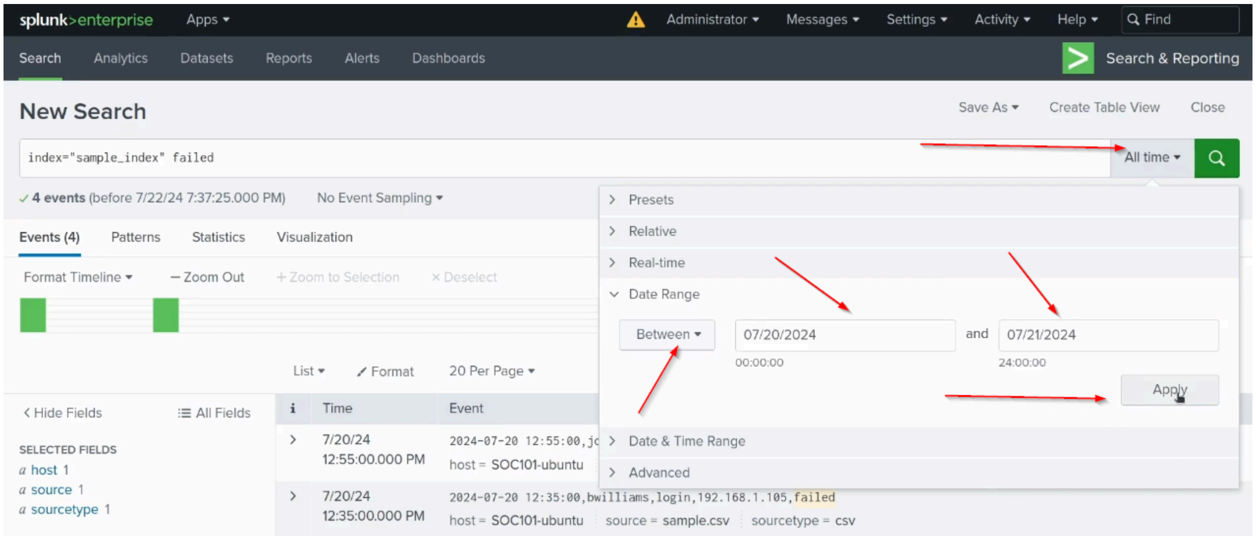
✓ 4 events (before 7/22/24 7:44:59.000 PM)

SELECTED FIELDS
- a host 1
- a source 1
- a sourcetype 1

INTERESTING FIELDS
- a authentication_status 1
- # date_hour 1
- # date_mday 1
- # date_minute 4
- a date_month 1
- # date_second 1
- a date_wday 1

Time	Event
7/20/24 12:55:00.000 PM	2024-07-20 12:55:00,jdoe,login,192.168.1.101,failed host = SOC101-ubuntu source = sample.csv sourcetype = csv
7/20/24 12:35:00.000 PM	2024-07-20 12:35:00,bwilliams,login,192.168.1.105,failed host = SOC101-ubuntu source = sample.csv sourcetype = csv
7/20/24 12:15:00.000 PM	2024-07-20 12:15:00,mjohnson,login,192.168.1.103,failed host = SOC101-ubuntu source = sample.csv sourcetype = csv
7/20/24 12:10:00.000 PM	2024-07-20 12:10:00,jdoe,login,192.168.1.101,failed host = SOC101-ubuntu source = sample.csv sourcetype = csv

Date and Time set:



New Search

```
index="sample_index" failed
```

All time ▾

✓ 4 events (before 7/22/24 7:37:25.000 PM)

Date Range: Between ▾ 07/20/2024 00:00:00 and 07/21/2024 24:00:00 Apply

SELECTED FIELDS
- a host 1
- a source 1
- a sourcetype 1

Time	Event
7/20/24 12:55:00.000 PM	2024-07-20 12:55:00,jc host = SOC101-ubuntu
7/20/24 12:35:00.000 PM	2024-07-20 12:35:00,bwilliams,login,192.168.1.105,failed host = SOC101-ubuntu source = sample.csv sourcetype = csv

🔍 What is Keyword Search in Splunk?

Keyword search in Splunk is the **basic and most common type of search** used to find specific words or phrases in your logs and indexed data — **without specifying any fields**.

🔍 Quote Search

Quote Search in Splunk is used to search for an **exact phrase**, including spaces or special characters, by enclosing it in **double quotes** `" "`.

Search	Description
`index="sample_index" "jdoe,login"`	Exact phrase match: **"jdoe,login"**
`index="sample_index" jdoe login`	Events containing **both** keywords anywhere
`index="sample_index" jdoe AND login`	Logical AND – both terms must exist

Search	Description
`index="sample_index" jdoe OR login`	Logical OR – either term can exist

🧨 Wildcard Search

Search	Matches
index="http_sample" `fail*`	fail, failed, failure, etc.
index="http_sample” `*004`	Ends with 004 (e.g., 2004, E004)
index="http_sample” `log*n`	login, logon, loggedin
index="http_sample” `12:*`	Any value starting with hour 12
index="http_sample” `12:*:00`	Values ending in `:00` during hour 12

🔠 Case Sensitivity

Search	Notes
index="http_sample” `failed`	Matches lowercase "failed"
`FAILED`	Matches uppercase "FAILED"

🧾 Field-Based Search

Operator	Search Example	Meaning
Equals `=`	`file="login.php"`	Exact match
Not equals `!=`	`file!="index.php"`	Excludes matches
Greater than `>`	`status>200`	Values greater than 200
Greater than or equal `>=`	`status>=404`	404 or above
Less than `<`	`status<500`	Below 500
Less than or equal `<=`	`status<=302`	302 and below

⚙️ Boolean Logic

Search	Explanation
`index="http_index" AND status>=200`	Only events with `status >= 200`

Search	Explanation
`index="http_index" OR status>=200`	All events from index, plus those with `status >= 200`
`index="http_index" AND method=GET OR method=POST`	Evaluates as: `(index="http_index" AND method=GET) OR method=POST`
`index="http_index" AND NOT method=GET OR method=POST`	Evaluates as: `(index="http_index" AND (NOT method=GET)) OR method=POST`

> 🔶 Order of Evaluation: NOT → OR → AND
>
> Use **parentheses** to control logic grouping

🧠 Using Parentheses

Search	Explanation
`index="http_sample" AND NOT (method=POST OR method=GET)`	Excludes events with `POST` or `GET` methods

🌐 IP/Client Matching

Search	Explanation
`clientip=100.*.*.*`	Wildcard match for all IPs starting with 100

🕓 Time Range Filtering

Search	Time Range
`earliest="07/17/2024:00:00:00" latest=now`	From specific time to current time
`earliest="07/17/2024:00:00:00" latest="07/17/2024:18:48:20"`	Specific time window

📁 Use Case: Apache Log Analysis

You can combine fields and time with wildcards and Boolean logic for deeper analysis.

Example:

```
spl
CopyEdit
index="http_index" file="access.log" status>=400 clientip=100.*.*.* method=
GET earliest="07/17/2024:00:00:00" latest=now
```

2.) Splunk - Search Commands

✅ What are Splunk Search Commands?

Splunk Search Commands are special instructions used in Splunk to:

- **Search** through logs and events
- **Filter**, **format**, and **analyze** data
- **Visualize** patterns and trends
- **Detect anomalies** or **summarize activity**

They are part of **SPL** (Search Processing Language), which powers how Splunk retrieves and processes data.

🧠 Why are they important?

- Help you **find exactly what you need** in huge log datasets
- Allow **real-time and historical analysis**
- Support **security monitoring**, **IT troubleshooting**, and **data reporting**

✅ SPLUNK IMPORTANT COMMANDS — FULL EXPLANATION

🔽 1. `sort`

```
spl
CopyEdit
index="http_sample" | sort -req_time
```

Use: Sorts results by `req_time` in **descending** order.

Why: To see slowest requests first (useful for performance analysis).

🔢 2. `stats count by clientip`

```
spl
CopyEdit
index="http_sample" | stats count by clientip
```

Use: Counts the number of events for each IP.

Why: To find **frequent visitors**, brute force attempts, or scanners.

🔝 3. `sort` + `stats` (most active IPs)

```
spl
CopyEdit
index="http_sample" | stats count by clientip | sort -count
```

Use: Sorts IPs by number of requests.

Why: To detect top talkers, scanning tools, or potential attacks.

🔟 4. `head`

```
spl
CopyEdit
index="http_sample" | stats count by clientip | sort -count | head 10
```

Use: Shows **top 10** most active IPs.

Why: Focus on the biggest requesters.

🔚 5. `tail`

```
spl
CopyEdit
index="http_sample" | stats count by clientip | sort count | tail 10
```

Use: Shows **least active IPs**.

Why: To catch rare or one-time probes.

📋 6. `table`

```
spl
CopyEdit
index="http_sample" | table _time, clientip, method, uri, useragent
```

Use: Displays selected fields in table format.

Why: Cleaner and easier to read/export.

🔁 7. `dedup`

```
spl
CopyEdit
index="http_sample" | table _time, clientip, method, uri, useragent | dedup useragent
```

Use: Shows only one event per `useragent`.

Why: Identify **unique user agents** (custom tools, scanners).

✏️ 8. `rename`

```
spl
CopyEdit
index="http_sample" | table _time, clientip, method, uri, useragent | rename useragent as "User Agent"
```

Use: Renames a field for readability.

Why: For dashboards or reporting.

📊 9. `top`

```
spl
CopyEdit
index="http_sample" | top limit=5 useragent
```

Use: Shows **most common** values of `useragent`.

Why: Identify popular browsers or scanning tools.

🧊 10. `rare`

```
spl
CopyEdit
index="http_sample" | rare limit=5 useragent
```

Use: Shows **least common** values of `useragent`.

Why: Spot stealthy or suspicious user agents.

📈 11. `chart`

```
spl
CopyEdit
index="http_sample" | chart count by status
```

Use: Group count of events by `status` (200, 404, 500, etc.).

Why: Helps see errors or unusual status codes.

⏱️ 12. `timechart`

```
spl
CopyEdit
index="http_sample" clientip="62.122.201.246" | timechart span=1s count
```

Use: Visualizes request activity over time for one IP.

Why: Detect **scanning patterns** or spikes.

🔍 13. `search` (filter specific value)

```
spl
CopyEdit
index="http_sample" clientip="62.122.201.246"
```

```
| table _time, clientip, useragent
| search useragent=*Nmap*
```

Use: Filter for events where user agent contains "Nmap".

Why: Detect **network scans** or attacks.

🌐 14. `iplocation`

```
spl
CopyEdit
index="http_sample" | iplocation clientip
```

Use: Adds geo fields: `city`, `country`, `region`, `lat`, `lon`.

Why: Trace where IPs come from — useful in **threat hunting**.

```
spl
CopyEdit
index="http_sample" | iplocation clientip | table _time, clientip, Country, City, uri
```

🗺️ 15. `geostats`

✅ Must use lowercase `country`:

```
spl
CopyEdit
index="http_sample" | iplocation clientip | geostats count by country
```

Use: Shows counts by country on a map.

Why: Visualize attacker/source IP **locations globally**.

📌 Summary Table of Commands

Command	Use / Purpose
`sort`	Order results by a field
`stats`	Group and summarize data
`head` / `tail`	Top or bottom N results
`table`	Show only selected fields
`dedup`	Remove duplicate values
`rename`	Rename a field for display
`top`	Show most frequent values
`rare`	Show least frequent values
`chart`	Count by a field (bar chart)
`timechart`	Count over time
`search`	Filter for matching strings
`iplocation`	Add geo fields from IP
`geostats`	Visual map by country/location

CTF Reference:

⇒ https://github.com/Sean-Everett/Splunk-Boss_of_the_SOC_v1