

Use Case 1: Failed Login Detection

1. Monitor failed login attempts

Do:

```
index=apache sourcetype=access_combined status=200 method=POST
```

```
| table_time clientip method uri_path status
```

Result: เพื่อใช้ในการ monitor ตรวจสอบการ login ผิดพลาดที่กำลังเข้ามาสู่ระบบ

New Search

Save As

Create Table View

Close

index=apache sourcetype=access_combined status=200 method=POST

table _time clientip method uri_path status

Last 24 hours

3 events (1/15/25 9:00:00.000 PM to 1/16/25 9:38:33.000 PM)

No Event Sampling

Job

Smart Mode

Events

Patterns

Statistics (3)

Visualization

20 Per Page

Format

Preview

_time	clientip	method	uri_path	status
2025-01-16 21:25:50	192.168.111.1	POST	/index.php	200
2025-01-16 21:25:32	192.168.111.1	POST	/index.php	200
2025-01-16 21:25:15	192.168.111.1	POST	/index.php	200

2. Track suspicious login patterns

Do:

```
index=apache sourcetype=access_combined status=200 method=POST clientip=192.168.111.128
```

```
| table_time clientip method uri_path status
```

เพื่อตรวจสอบการขาด pattern ที่ดูเหมือนเป็นภัยคุกคาม

[illegible]

3. Set alerts for brute force attempts

Do:

The screenshot shows the Splunk Enterprise interface. The search bar contains the query: `index=apache sourcetype=access_combined status=200 method=POST clientip=192.168.111.128 | table _time clientip method uri_path status`. The search results show 30,633 events. The table view displays columns for _time, clientip, method, uri_path, and status. The status column shows 200 for all events. The uri_path column shows /index.php for all events. The method column shows POST for all events. The clientip column shows 192.168.111.128 for all events. The _time column shows a range of timestamps from 2025-01-16 22:08:06 to 2025-01-16 22:08:06.

_time	clientip	method	uri_path	status
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200
2025-01-16 22:08:06	192.168.111.128	POST	/index.php	200

The screenshot shows the Splunk Enterprise interface. The search bar contains the query: `index=apache sourcetype=access_combined method=POST status=200 | stats count by clientip | where count > 10`. The search results show 30,649 events. The table view displays columns for clientip and count. The clientip column shows 192.168.111.1 and 192.168.111.128. The count column shows 16 and 30633 respectively.

clientip	count
192.168.111.1	16
192.168.111.128	30633

- Setting alert ตาม method ที่ user ต้องการ

Save As Alert

Settings

Title

Suspicious Login Activity (POST 200)

Description

For implement and testing alert login failed > 10 events/5 min

Permissions

Private

Shared in App

Alert type

Scheduled

Real-time

Run on Cron Schedule

Time Range

Last 24 hours

Cron Expression

*/5 * * * *

e.g. 00 18 *** (every day at 6PM). [Learn More](#)

Expires

24

hour(s)

Trigger Conditions

Trigger alert when

Custom

search count > 10

e.g. "search count > 10". Evaluated against the results of the base search.

Trigger

Once

For each result

Throttle

Trigger Actions

+ Add Actions

When triggered

Add to Triggered Alerts

Remove

Severity

Medium

Cancel

Save

splunk>enterprise

Apps

Administrator

Messages

Settings

Activity

Help

Find

Search

Analytics

Datasets

Reports

Alerts

Dashboards

Alerts

Alerts set a condition that triggers an action, such as sending an email that contains the results of the triggering search to a list of people. Click the name to view the alert. Open the alert in Search to refine the parameters.

1 Alerts

All

Yours

This App's

filter

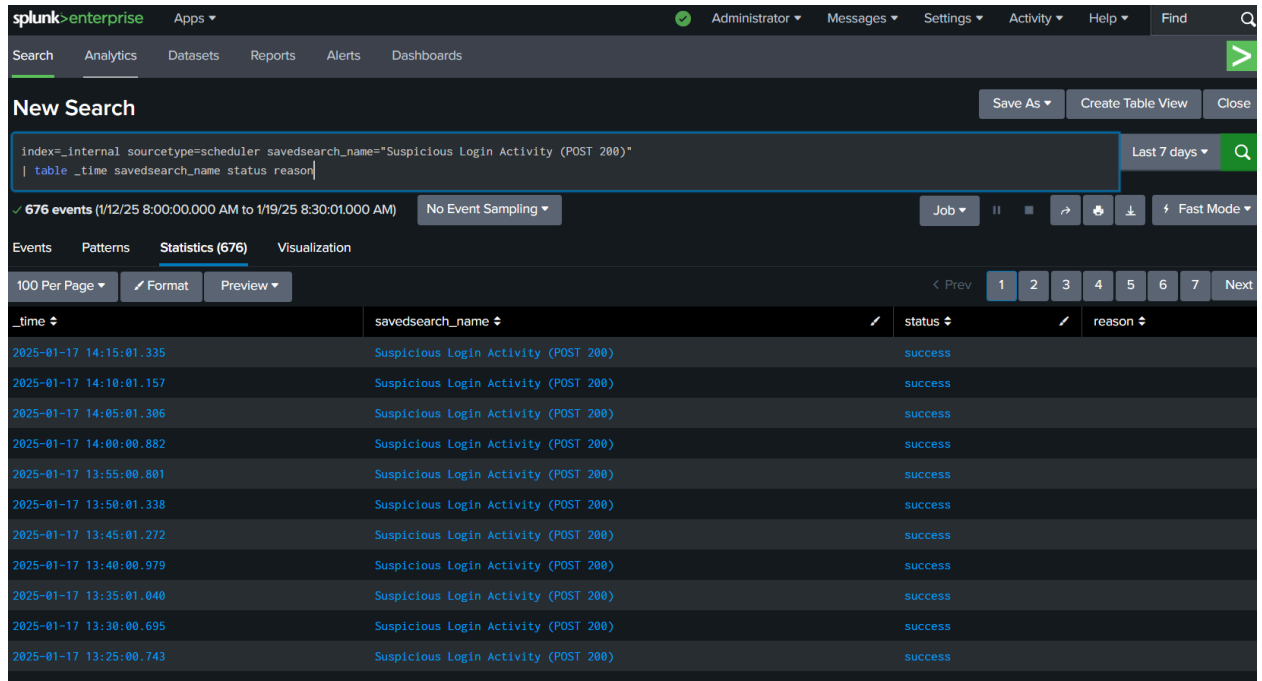
i	Title ^	Actions	Owner	App	Sharing	Status
▼	<div>Suspicious Login Activity (POST 200)</div> <div>For implement and testing alert login failed > 10 events/5 min</div> <div> <div>Enabled:</div> <div>Yes. Disable</div> </div> <div> <div>Permissions:</div> <div>Private. Owned by splunkadmin. Edit</div> </div> <div> <div>Modified:</div> <div>Jan 17, 2025 12:04:50 AM</div> </div> <div> <div>Alert Type:</div> <div>Scheduled. Cron Schedule. Edit</div> </div> <div> <div>Trigger Condition: ..</div> <div>Custom. "search count > 10". Edit</div> </div> <div> <div>Actions:</div> <div>▼ 1 Action Edit</div> <div>Add to Triggered Alerts</div> </div>	<div>Open in Search</div> <div>Edit</div>	splunkadmin	search	Private	Enabled

Do:

index=_internal sourcetype=scheduler savedsearch_name="Suspicious Login Activity (POST 200)"

| table _time savedsearch_name status reason

Result:



The screenshot displays the Splunk Enterprise web interface. At the top, the navigation bar includes 'splunk>enterprise', user 'Administrator', and various menu items like 'Messages', 'Settings', 'Activity', 'Help', and 'Find'. Below this, a secondary navigation bar shows 'Search', 'Analytics', 'Datasets', 'Reports', 'Alerts', and 'Dashboards'. The main content area is titled 'New Search' and contains a search bar with the query: `index=_internal sourcetype=scheduler savedsearch_name="Suspicious Login Activity (POST 200)" | table _time savedsearch_name status reason`. To the right of the search bar are buttons for 'Save As', 'Create Table View', and 'Close'. Below the search bar, a status bar indicates '676 events (1/12/25 8:00:00.000 AM to 1/19/25 8:30:01.000 AM)' and a 'No Event Sampling' dropdown. Below this, there are tabs for 'Events', 'Patterns', 'Statistics (676)', and 'Visualization'. The 'Statistics (676)' tab is active, showing a table with 100 rows per page. The table has four columns: '_time', 'savedsearch_name', 'status', and 'reason'. The data shows multiple entries for 'Suspicious Login Activity (POST 200)' with a status of 'success'.

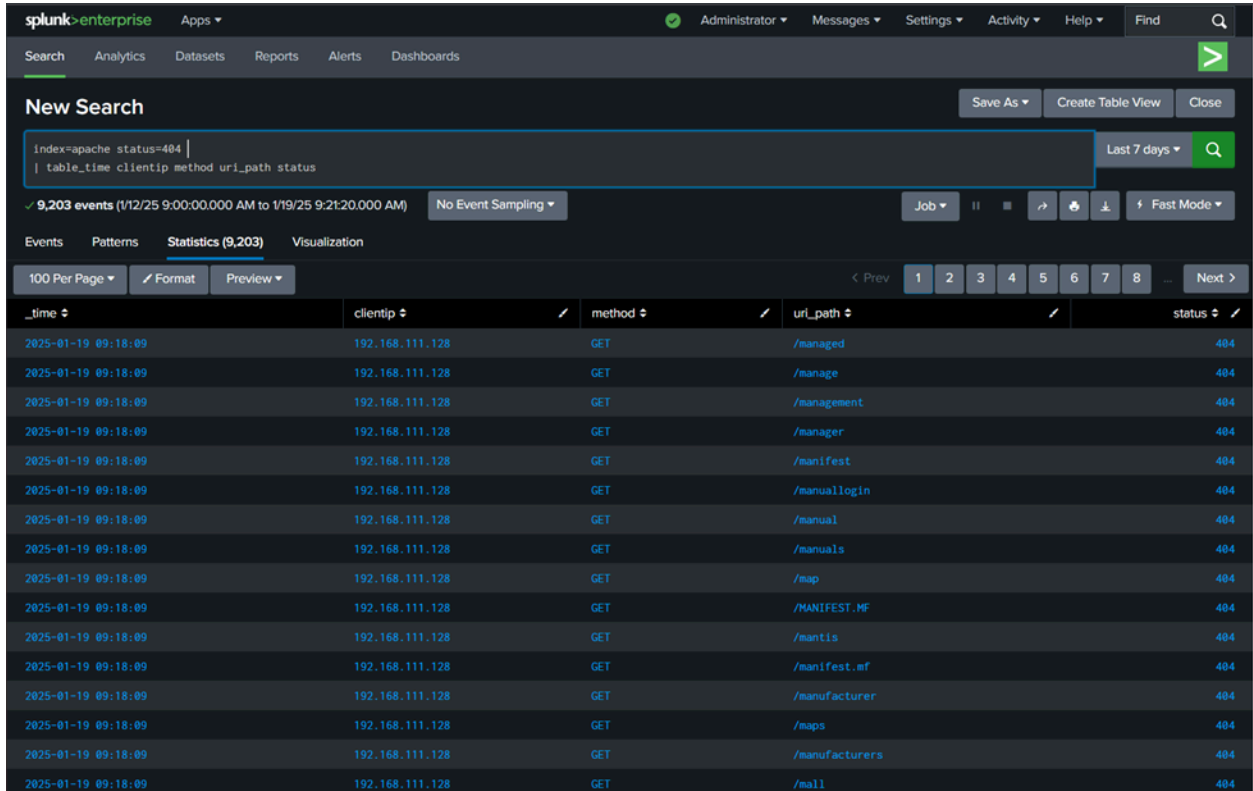
_time	savedsearch_name	status	reason
2025-01-17 14:15:01.335	Suspicious Login Activity (POST 200)	success	
2025-01-17 14:10:01.157	Suspicious Login Activity (POST 200)	success	
2025-01-17 14:05:01.306	Suspicious Login Activity (POST 200)	success	
2025-01-17 14:00:00.882	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:55:00.801	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:50:01.338	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:45:01.272	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:40:00.979	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:35:01.040	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:30:00.695	Suspicious Login Activity (POST 200)	success	
2025-01-17 13:25:00.743	Suspicious Login Activity (POST 200)	success	

Use Case 2: Unauthorized Directory/Path Access Detection

1. Detect finding paths or hidden files on server

Do: index=apache status=404

Result:



The screenshot displays the Splunk Enterprise web interface. At the top, the navigation bar includes 'splunk>enterprise', 'Apps', and user options like 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and 'Find'. Below this is a 'New Search' section with a search bar containing the query 'index=apache status=404 | table _time clientip method uri_path status'. The search results show 9,203 events from 1/12/25 to 1/19/25. The 'Statistics (9,203)' tab is selected, showing a table of events. The table has columns for _time, clientip, method, uri_path, and status. All events show a status of 404, indicating unauthorized access to various paths like /managed, /manage, /management, /manager, /manifest, /manuallogin, /manual, /manuals, /map, /MANIFEST.MF, /mantis, /manifest.mf, /manufacturer, /maps, /manufacturers, and /mail.

_time	clientip	method	uri_path	status
2025-01-19 09:18:09	192.168.111.128	GET	/managed	404
2025-01-19 09:18:09	192.168.111.128	GET	/manage	404
2025-01-19 09:18:09	192.168.111.128	GET	/management	404
2025-01-19 09:18:09	192.168.111.128	GET	/manager	404
2025-01-19 09:18:09	192.168.111.128	GET	/manifest	404
2025-01-19 09:18:09	192.168.111.128	GET	/manuallogin	404
2025-01-19 09:18:09	192.168.111.128	GET	/manual	404
2025-01-19 09:18:09	192.168.111.128	GET	/manuals	404
2025-01-19 09:18:09	192.168.111.128	GET	/map	404
2025-01-19 09:18:09	192.168.111.128	GET	/MANIFEST.MF	404
2025-01-19 09:18:09	192.168.111.128	GET	/mantis	404
2025-01-19 09:18:09	192.168.111.128	GET	/manifest.mf	404
2025-01-19 09:18:09	192.168.111.128	GET	/manufacturer	404
2025-01-19 09:18:09	192.168.111.128	GET	/maps	404
2025-01-19 09:18:09	192.168.111.128	GET	/manufacturers	404
2025-01-19 09:18:09	192.168.111.128	GET	/mail	404

Use Case 3: Privilege Escalation Detection

Monitor and detect unauthorized privilege elevation

ตรวจสอบ Privilege Escalation ด้วย EventCode (4672, 4732, 4728)

Do:

```
source="splunk_logs_wineventlog.csv"
| rex field=_raw "EventCode=(?<EventCode>\d+)"
| rex field=_raw "LogName=(?<LogName>[^\,]+)"
| rex field=_raw "Message=(?<Message>[^\,]+)"
| where EventCode IN ("4672", "4732", "4728")
| table _time EventCode LogName Message
```

Result:

The screenshot shows a Splunk search interface with the following search query:

```
source="splunk_logs_wineventlog.csv"
| rex field=_raw "EventCode=(?<EventCode>\d+)"
| rex field=_raw "LogName=(?<LogName>[^\,]+)"
| rex field=_raw "Message=(?<Message>[^\,]+)"
| where EventCode IN ("4672", "4732", "4728")
| table _time EventCode LogName Message
```

The search results show 11 events. The first event is expanded, showing the following details:

Field	Value
EventCode	4672
LogName	Security
Message	Special privileges assigned to new logon.
Subject:	Security ID: S-1-5-18 Account Name: SYSTEM Account Domain: NT AUTHORITY Logon ID: 0x3E7
Privileges:	SeAssignPrimaryTokenPrivilege SeTcbPrivilege SeSecurityPrivilege SeTakeOwnershipPrivilege SeLoadDriverPrivilege SeBackupPrivilege SeRestorePrivilege SeDebugPrivilege SeAuditPrivilege SeSystemEnvironmentPrivilege

Alert Conditions:

- 1) Addition of users to privileged groups
- 2) Special privileges assigned to standard users
- 3) Multiple privilege changes in short timeframe

Response Actions:

- 1) Immediate notification to security team
- 2) Review of privilege changes
- 3) Rollback unauthorized changes

Use Case 4: Service Change Monitoring

Detect unauthorized service modifications

ตรวจสอบ Service Changes (7045, 7040, 7036):

Do:

```
source="splunk_logs_wineventlog.csv"
| rex field=_raw "EventCode=(?<EventCode>\d+)"
| rex field=_raw "LogName=(?<LogName>[^\,]+)"
| rex field=_raw "Message=(?<Message>[^\,]+)"
| where EventCode IN ("7045", "7040", "7036")
| table _time EventCode LogName Message
```

Result:

The screenshot displays the Splunk search interface. At the top, the search bar contains the following query:

```
source="splunk_logs_wineventlog.csv"
| rex field=_raw "EventCode=(?<EventCode>\d+)"
| rex field=_raw "LogName=(?<LogName>[^\,]+)"
| rex field=_raw "Message=(?<Message>[^\,]+)"
| where EventCode IN ("7045", "7040", "7036")
| table _time EventCode LogName Message
```

Below the search bar, the results are shown in a table format. The table has four columns: **_time**, **EventCode**, **LogName**, and **Message**. There are two events displayed.

_time	EventCode	LogName	Message
2024-12-24 09:49:39	7036	System	The Network Setup Service service entered the stopped state."
2024-12-24 09:50:00	7036	System	The Software Protection service entered the running state."

The first event shows a service state change for the Network Setup Service. The second event shows a service state change for the Software Protection service. Both events have an EventCode of 7036 and are categorized as System logs.

Alert Conditions:

- 1) Critical services stopped
- 2) Multiple service changes in short period
- 3) Service changes outside maintenance windows
- 4) New service installation

Response Actions:

- 1) Alert system administrators
- 2) Compare against change management records
- 3) Review service configurations

Use Case 5: Suspicious Database Access Detection

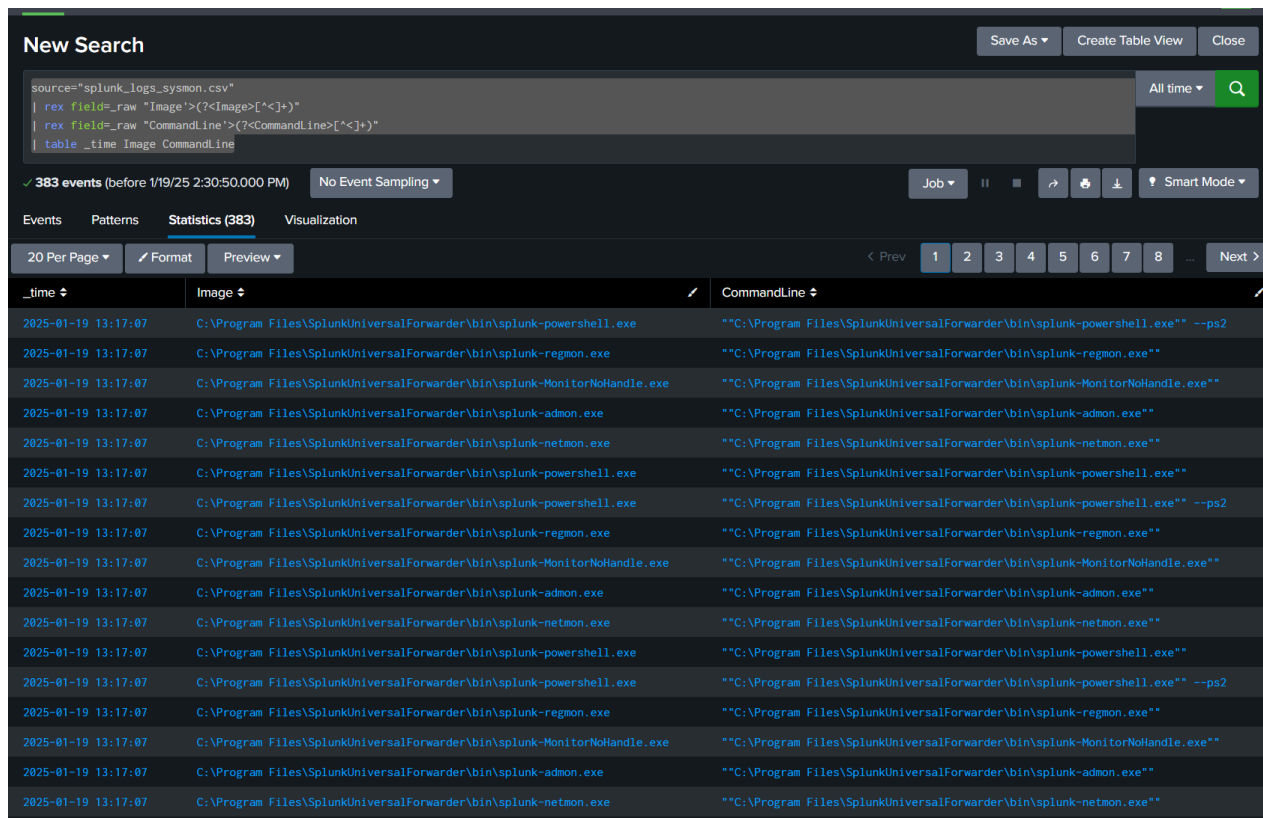
Monitor and detect unauthorized database access

ตรวจสอบ Splunk_logs_sysmon และ ตรวจสอบ Process Creation:

Do:

```
source="splunk_logs_sysmon.csv"
| rex field=_raw "Image">{<Image>[^<]+}"
| rex field=_raw "CommandLine">{<CommandLine>[^<]+}"
| table _time Image CommandLine
```

Result:



The screenshot shows the Splunk Search interface with a search query that filters for process creation events from the 'splunk_logs_sysmon.csv' source. The query uses regular expressions to extract the 'Image' and 'CommandLine' fields from the raw data. The search results are displayed in a table with columns for time, image path, and command line. The results show various system processes being executed, including powershell.exe, regmon.exe, MonitorNoHandle.exe, admon.exe, and netmon.exe, all running from the Splunk Universal Forwarder bin directory.

_time	Image	CommandLine
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe" --ps2
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe" --ps2
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-powershell.exe" --ps2
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-regmon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-MonitorNoHandle.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-admon.exe"
2025-01-19 13:17:07	C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe	"C:\Program Files\SplunkUniversalForwarder\bin\splunk-netmon.exe"

ตรวจสอบ MySQL commands

Do:

```
source="splunk_logs_sysmon.csv"
| rex field=_raw "CommandLine'>(?<CommandLine>[^\<]+)"
| search CommandLine="*select*"
| table _time CommandLine
```

Result:

The screenshot shows the Splunk search interface. At the top, the search bar contains the following query:

```
source="splunk_logs_sysmon.csv"
| rex field=_raw "CommandLine'>(?<CommandLine>[^\<]+)"
| search CommandLine="*select*"
| table _time CommandLine
```

Below the search bar, the results are displayed in a table. The table has two columns: **_time** and **CommandLine**. The first row shows the following data:

_time	CommandLine
2025-01-19 13:17:07	"C:\xampp\mysql\bin\mysql.exe" -u root --password= -e "use kbtg; select * from flag;"

Alert Conditions:

- 1) Root login without password
- 2) Access to sensitive tables
- 3) Unusual query patterns
- 4) High volume of SELECT statements

Response Actions:

- 1) Log suspicious activities
- 2) Review database access logs
- 3) Implement additional access controls