🔍 SPL (Search Processing Language) — SOC Analyst Core

First: What SPL *really* is

SPL is not “programming.”
It’s a **pipeline** that:

1. **Finds events**
2. **Transforms them**
3. **Summarizes behavior**

Think:

```
SEARCH → FILTER → GROUP → COUNT → DECIDE
```

Phase A — MUST-KNOW SPL COMMANDS (Non-Negotiable)

These 7 commands account for **80%+ of SOC searches**.

1️⃣ `search` (Implicit, but critical)

What it does

Filters events.

Example

```
index=windows_logs EventCode=4625
```

Meaning:

> “Find failed Windows logins.”

You’re already using it correctly.

🔹 **SOC usage**

- Initial triage
- Narrowing alerts
- Validation

2️⃣ `table` — Make logs human-readable

What it does

Selects specific fields and formats output.

Example

```
index=windows_logs EventCode=4625
| table _time host user SourceIp
```

🔹 **SOC usage**

- Incident timelines
- Reports
- Screenshots for evidence

💡 SOC rule:

> If you can’t explain it, `table` it.

3️⃣ `stats` — The most important command

What it does

Aggregates events.

What does `stats` mean?

`stats` **does NOT** mean:

✅ **`stats` = statistics**

It is used to **summarize many events into numbers**.

What does `| stats count` do?

```
| stats count
```

What it does

- Counts **how many events** matched your search
- Outputs **ONE result**

Example

```
index=windows_logs EventCode=4625

| stats count
```

Output:

```
count = 37
```

Meaning:

> “There were 37 failed logins.”

❗ Important:

- It does **NOT** show events
- It does **NOT** show time
- It **destroys individual event details**

Once you run `stats`, you’re no longer looking at raw logs.

What does `| stats count by host` do?

```
| stats count by host
```

Meaning

- Count events **per host**

Example output:

```
host            count

WIN11-LAB-01    25

UBUNTU-LAB-01   12
```

Meaning:

> “Windows had 25 events, Ubuntu had 12.”

What does `| stats count by user SourceIp` do?

```
| stats count by user SourceIp
```

This creates **groups**:

user	SourceIp	count
admin	10.0.0.5	12
guest	10.0.0.9	2

Meaning:

> “This user tried to log in from this IP X times.”

❗ Important:

- It shows **how many times each pair appears**

Common forms

```
| stats count
| stats count by host
| stats count by user SourceIp
```

Example

```
index=windows_logs EventCode=4625
| stats count by user SourceIp
```

Meaning:

> “Who is failing logins and from where?”

🔹 **SOC usage**

- Brute force detection
- Activity summarization
- Alert thresholds

⚠️ If you learn **only one SPL command well**, make it `stats`.

4️⃣ `sort` — Prioritize what matters

What it does

Orders results.

Example

```
| sort -count
```

Meaning:

> Highest activity first.

🔹 **SOC usage**

- Spot worst offenders
- Focus investigation quickly

5️⃣ `where` — Apply logic

What it does

Filters after aggregation.

What does `| where count > 5` mean?

```
| where count > 5
```

Meaning:

> “Only show groups where the event happened MORE than 5 times.”

Example:

- 2 failures → hidden
- 6 failures → shown

Example

```
| where count > 5
```

Meaning:

> “Show only suspicious volumes.”

🔹 **SOC usage**

- Threshold-based detections
- Noise reduction

6️⃣ `timechart` — SOC's favorite visualization

What it does

Shows trends over time.

Example

```
index=windows_logs EventCode=4625
| timechart count
```

🔹 **SOC usage**

- Attack timelines
- Spikes & anomalies
- Dashboards

7️⃣ `bucket` — Time grouping (detection logic)

What it does

Groups events into time windows.

What does `| bucket _time span=5m` do?

```
| bucket _time span=5m
```

❌ It does NOT filter events
✅ It **groups time into 5-minute windows**

Example timestamps:

```
10:01
```

```
10:03
```

```
10:04
```

Become:

```
10:00–10:05
```

Example

```
| bucket _time span=5m
```

This enables:

> “X events within Y minutes”

Phase B — CORE SOC DETECTION PATTERNS

You **do not memorize queries** — you memorize **patterns**.

🔴 Pattern 1 — Brute Force Login

```
index=windows_logs EventCode=4625
| bucket _time span=5m
| stats count by _time host user
| where count > 5
```

Meaning:

> “More than 5 failed logins in 5 minutes.”

💼 Real SOC use:

- Alert
- Investigation
- Ticket creation

Why this order matters (VERY IMPORTANT)

```
index=windows_logs EventCode=4625
| bucket _time span=5m
| stats count by _time host user
| where count > 5
```

YES — ORDER MATTERS

Because:
1️⃣`bucket` must happen **before** `stats`
2️⃣`where` must happen **after** `stats`

❌ This would break:

```
| where count > 5
| stats count by user
```

Because `count` doesn’t exist yet.

Think:

```
Create → Aggregate → Filter
```

🔴 Pattern 2 — Rare / Suspicious Process Execution

What is `Image`?

```
| stats count by Image
```

Image is NOT a picture ❌

`Image` =
👉 **Executable path**

Example:

```
C:\Windows\System32\cmd.exe
C:\Users\Public\evil.exe
```

```
index=windows_logs
| stats count by Image
| sort count
```

Meaning:

> “What runs rarely?”

🔹 Rare ≠ malicious
🔹 Rare = worth investigating

Why SOCs use `Image`

Processes = behavior.

Rare process = suspicious.

```
| stats count by Image
| sort count
```

Low count = rare
Rare = investigate

This is **behavior-based detection**.

🔴 Pattern 3 — New Admin Activity

```
index=security_logs EventCode=4672
| stats count by user host
```

Meaning:

> "Who logged in with special privileges?"

Used for:

- Privilege escalation detection
- Insider threat monitoring

Phase C — FIELDS YOU MUST UNDERSTAND

This matters **more than EventCodes**.

🔑 Core Fields (Memorize These)

Field	Meaning
`_time`	When event occurred
`host`	Machine generating event
`user`	Account involved
`SourceIp`	Origin of activity

`Image`	Executable path
`CommandLine`	How process was run
`sourcetype`	Log format
`index`	Log storage category

SOC analysts think in:

> **WHO → FROM WHERE → DID WHAT → WHEN**

Example: Full Investigation Query

```
index=windows_logs EventCode=4625
| table _time host user SourceIp
| sort _time
```

This answers:

- Who?
- From where?
- When?
- On which system?

That’s **incident response**.

SOC Mental Model (MEMORIZE THIS)

```
search → bucket (optional) → stats → where → table → sort
```