Task 2 — Security Alert Monitoring & Incident Response

Project: Security Alert Monitoring & Incident Response (Future Interns — CS Task 2)

Student: AKASH M

Date: 26-09-2025

_

Tools: Splunk (Free), Sample logs (EVTX/CSV/JSON), Windows PowerShell / python-evtx

---

1. Summary

A brief 2–3 line summary of what you did (upload logs, created searches/alerts, documented incidents).

Example: Uploaded Windows event samples to Splunk, created searches to detect brute-force and SQLi attempts, configured alerts, and documented 3 incidents with remediation steps.

---

2. Environment & Data

- **SIEM / Tool used:** *Splunk Enterprise (Free Trial)*
- **Index created:** `security_logs` (or `main`)
- **Sample log files used:**
  - `sample.evtx` → converted to `sample.csv` (if applicable)
  - `web_access.log` (apache)
  - [List other files used]
- **Conversion / parsing steps:**
  - If EVTX → used PowerShell: `Get-WinEvent -Path .\sample.evtx | Export-Csv -Path .\sample.csv -NoTypeInformation`
  - Sourcetype set to: `csv` / `winlog` / `apache:access`

---

3. Objectives

1. Ingest and parse security logs into Splunk.
2. Create searches to identify suspicious activity (failed logins, SQLi, new admin creation).
3. Configure alerts for high-severity events.

4. Document incidents with evidence, classification and remediation.

---

4. Key Searches (copy-paste into Splunk)

Failed SSH (brute force)

```
index=security_logs "Failed password" OR "Failed login"
| stats count by src_ip user host
| where count>5
| sort -count
```

SQL Injection patterns (web logs)

```
index=security_logs sourcetype=apache:access OR sourcetype=nginx:access
| search " OR " "1=1" "' OR '1'='1"
| table _time clientip request status uri query
```

New user / admin creation (Windows Event IDs)

```
index=security_logs (EventID=4720 OR EventID=4728 OR EventID=4672)
| table _time host user EventID Message
```

---

5. Alerts Configured

- **Alert A — Brute-force SSH**
  - Search: *Failed SSH search above*
  - Frequency: Scheduled every 5 minutes
  - Trigger: If any IP has `count>10` in 5m
  - Action: Email to admin@example.com, add to ticketing system
  - Throttle: 1 hour
- **Alert B — SQL Injection detected**
  - Search: *SQLi search above*
  - Frequency: Real-time (or every 1m)
  - Trigger: Any result found
  - Action: Post to Slack webhook

(Include screenshots of the Alert configuration here)

---

6. Findings (Incidents Summary)

Incident ID	Observed Time (UTC)	Type	Indicators	Severity	Status
INC-001	2025-09-24 10:15:23	Brute Force (SSH)	src_ip=192.168.1.50, user=root	Medium	Open
INC-002	2025-09-24 10:22:01	SQL Injection (Web)	clientip=192.168.1.75, payload `OR '1'='1`	High	Resolved
INC-003	2025-09-24 10:35:42	New Admin Creation	EventID=4720 user=Attacker	High	Open

---

7. Incident Details (Use one block per incident)

INC-001 — Brute Force (SSH)

- **Observed:** `2025-09-24 10:15:23`
- **Raw Evidence (copy of Splunk event):**

```
Sep 24 10:15:23 server1 sshd[1245]: Failed password for root from 192.168.1.50 port 54432 ssh2
Sep 24 10:15:28 server1 sshd[1245]: Failed password for root from 192.168.1.50 port 54433 ssh2
```

- **Indicators:** `src_ip=192.168.1.50`, repeated failed auths, >10 attempts in 5 minutes
- **Severity:** Medium (escalate to High if success observed)
- **Containment & Remediation:**
  1. Block `192.168.1.50` on firewall.
  2. Enable `fail2ban` / account lockout policies.
  3. Review `/var/log/auth.log` for lateral movement.
- **Root Cause:** Weak SSH authentication - no MFA for admin.
- **Status / Notes:** Open — awaiting firewall block.
- **Screenshots / Evidence:** `screenshot_inc001_dashboard.png` *(placeholder)*

INC-002 — SQL Injection (Web)

- **Observed:** `2025-09-24 10:22:01`
- **Raw Evidence:**

```
192.168.1.75 - - [24/Sep/2025:10:22:01 +0530] "GET /index.php?id=1' OR '1'='1 HTTP/1.1" 200 532
```

- **Severity:** High (attempt to manipulate DB)
- **Containment & Remediation:**
    1. Block malicious IP at WAF / firewall.
    2. Apply parameterized queries and input validation.
    3. Review database logs for suspicious queries / exfil.
- **Status / Notes:** Resolved (WAF rule added)
- **Screenshots / Evidence:** `screenshot_inc002_rawlog.png`, `screenshot_inc002_wafrule.png` *(placeholders)*

---

8. Recommendations (Short & Actionable)

1. Enforce MFA for all admin accounts.
2. Implement WAF rules and parameterized DB queries to stop SQLi.
3. Harden SSH (disable root login, use keys, setup fail2ban).
4. Regularly ingest Windows and network logs into SIEM for correlation.
5. Create runbook for incident triage & escalation.

Prepard by : AKASH M

Content : makash6420@gmail.com