Log Analysis: Detecting Suspicious Authentication Attempts

Project Overview

This project demonstrates my ability to analyze authentication logs, detect abnormal behavior, and think like a SOC (Security Operations Center) analyst. I created a sample log file in **JSON format** and investigated suspicious login activity that could indicate a brute-force attack or account compromise.

The goal was to simulate a real-world scenario, analyze the logs step by step, and document my process clearly — just like I would in a professional cybersecurity role.

Project Files

- **`auth_logs.json`** Contains the sample authentication logs.
- **`README.md`** Documentation of the project (this file).

Step 1: Reviewing the Logs

I created a log dataset in `auth_logs.json` with the following structure:

```
{
    "timestamp": "2025-08-25T08:01:12Z",
    "event_type": "authentication_failure",
    "username": "john.d",
    "source_ip": "192.168.1.44",
    "destination_host": "websrv-02",
    "protocol": "ssh",
    "action": "denied",
    "reason": "invalid_password"
}
```

Each log entry contains: - **When** it happened `timestamp` - **What** happened `event_type` - **Which user** `username` - **Where it came from** `source_ip` - **Where it was going** `destination_host` - **Action** allowed or denied

---

Step 2: Analyzing the Events

From the log file, I observed the following sequence:

1. `john.d` failed login from **192.168.1.44** (twice)

2. Failed login from a **different IP (45.77.22.10)**
3. Successful login from a **new IP (203.0.113.55)**

Step 3: Identifying Suspicious Behavior

- Normal users might mistype a password a few times, but always from the same IP (same device).
- Here, the failed attempts came from multiple different IPs.
- Then, a successful login occurred from yet another IP.

This strongly suggests a brute-force attack or credential stuffing, followed by a successful compromise.

Step 4: SOC Actions I Would Take

1. Containment Disable or reset the account `john.d` .
2. Block malicious IPs Add `45.77.22.10` and `203.0.113.55` to firewall blocklists.
3. Investigate further Check what actions `john.d` performed after login.
4. Notify the user and reset their password.

Step 5: Writing Detection Queries

To detect this behavior automatically, I wrote example queries.

Splunk Query

```
index=auth_logs username="john.d"
| stats count by event_type, source_ip
| eventstats count(eval(event_type="authentication_failure")) as failed_count
| eventstats values(eval(event_type="authentication_success")) as success
| where failed_count >= 3 AND success="authentication_success"
```

```
{
  "query": {
    "bool": {
```

ELK (Kibana Query DSL)

```
        "must": [
          { "match": { "username": "john.d" } },
          { "terms": { "event_type": ["authentication_failure",
"authentication_success"] } }
        ]
      }
    }
  }
```

These queries help detect when a user account has multiple failed logins from different IPs before a success.

Key Skills Demonstrated

- Log file creation in JSON format
- Log analysis & investigation
- Identifying brute-force attacks
- Writing Splunk & ELK queries
- Documenting security findings clearly

Why This Project Matters

This project shows my ability to: - Think like a SOC analyst - Investigate suspicious activity - Write detection queries - Document findings for both technical and non-technical audiences

I not only understand logs, but I also know how to turn raw data into actionable security insights.