🔐 AWK-Based Log Analysis for Security Monitoring & Reporting

📌 Overview

This project demonstrates how **AWK can be used for security-focused log analysis**, including filtering, aggregation, anomaly signal identification, and lightweight automated reporting from structured server logs.

The goal is to show how **native Linux tools** can support **SOC triage, incident investigation, and operational visibility**, particularly when centralized SIEM access is limited or unavailable.

🎯 Objective

- Analyze structured server logs using AWK
- Identify potentially risky request patterns
- Summarize error conditions and traffic trends
- Generate a lightweight, human-readable security report

🧩 Scenario

A system administrator or SOC analyst is tasked with reviewing server access logs to:

- Identify higher-risk request types (e.g., POST requests)
- Detect error patterns (4xx / 5xx responses)
- Understand traffic distribution by IP and requested resource
- Produce a concise operational or security summary

AWK is used as a **fast, scriptable alternative** during early-stage investigations or host-based triage.

🛠️ Tools & Technologies

- Linux (Bash)
- `awk`
- Core Unix utilities (head, pipes, redirection)
- Structured log data
  (timestamp, IP address, HTTP method, resource, status code)
- Basic HTML generation for reporting

🔍 Methodology

1️⃣Log Structure Validation

- Verified consistent log formatting
- Confirmed predictable field positions for reliable parsing

Security relevance:
Accurate parsing is critical before applying detection logic or aggregation.

2️⃣Field Extraction for Visibility

- Isolated IP addresses and requested resources
- Reduced noise by focusing on investigation-relevant fields

Security relevance:
Supports rapid identification of client behavior and targeted endpoints.

3️⃣ Conditional Filtering for Risk Signals

Applied conditions to surface:

- POST requests (higher risk than GET)
- Client and server error responses (404, 500, etc.)
- POST requests resulting in error codes

Security relevance:
POST + error patterns may indicate:

- Failed authentication attempts
- Application abuse
- Misconfigured endpoints
- Early indicators of exploitation

4️⃣ Aggregation & Trend Analysis

Used AWK associative arrays to:

- Count HTTP status codes
- Identify most frequently accessed resources
- Identify high-volume client IP addresses

Security relevance:
Helps distinguish normal traffic patterns from anomalies and abuse.

5️⃣ Automated Reporting

Created a reusable AWK script to generate an HTML summary report containing:

- Total request volume
- Error rate
- Top client IPs
- Most requested resources

Security relevance:
Demonstrates how analysts can transform raw logs into **actionable security summaries** without heavy tooling.

🧠 Key Findings

- Error responses (4xx / 5xx) are strong indicators of application stress or misuse
- POST requests require closer scrutiny due to authentication and data submission risks
- Request concentration by IP or resource may indicate:
    - Normal usage patterns
    - Misconfigurations
    - Potential abuse or reconnaissance