Engineering Skills Project: Web Server Log Analysis

Sherise Ee

July 2025

Problem Overview

A small music media startup is facing server slowdowns and downtime due to a spike in website traffic. With only 3 engineers on staff, this downtime is severely impacting their productivity. They suspect this is due to automated traffic (bots) overwhelming the server. A scalable and cost-effective technical solution is needed to identify the cause and mitigate future overloads.

Analysis Approach

Using Python, a script was developed to analyse the provided server access log. The script extracts and processes key fields such as IP address, timestamp, HTTP method, endpoint, status code, and response time. The program performs the following:

- Identifies the top IP addresses and most requested endpoints
- Flags suspicious behavior (repeated `POST` requests and 404 errors)
- Detects IPs with unusually high request rates (more than 5 requests per second)
- Saves suspicious IPs to a `blocklist.txt` file for future use
- Generates bar charts of top IPs and endpoints using `matplotlib`

Findings

The analysis revealed that a small number of IPs contributed to the majority of the traffic. Endpoints such as `/api/episodes` and `/subscribe-premium` were frequently targeted. Several IPs demonstrated bot-like behavior, such as high request frequency and repeated 404 responses.

Proposed Solution

A lightweight, automated mitigation strategy is recommended:

- **NGINX rate limiting:** restrict traffic to sensitive routes, e.g.

```
limit_req_zone $binary_remote_addr zone=api_limit:10m rate=1r/s;
location /api/ {
    limit_req zone=api_limit burst=5 nodelay;
}
```

- **CAPTCHA:** apply to POST endpoints e.g. login and subscriptions
- **Cloudflare (optional):** to block bots and cache static content

Conclusion

The log analysis tool successfully identifies abnormal traffic patterns and suggests low-cost, automated solutions. The entire system is containerized with Docker to deploy across environments easily.