Log Analysis Report

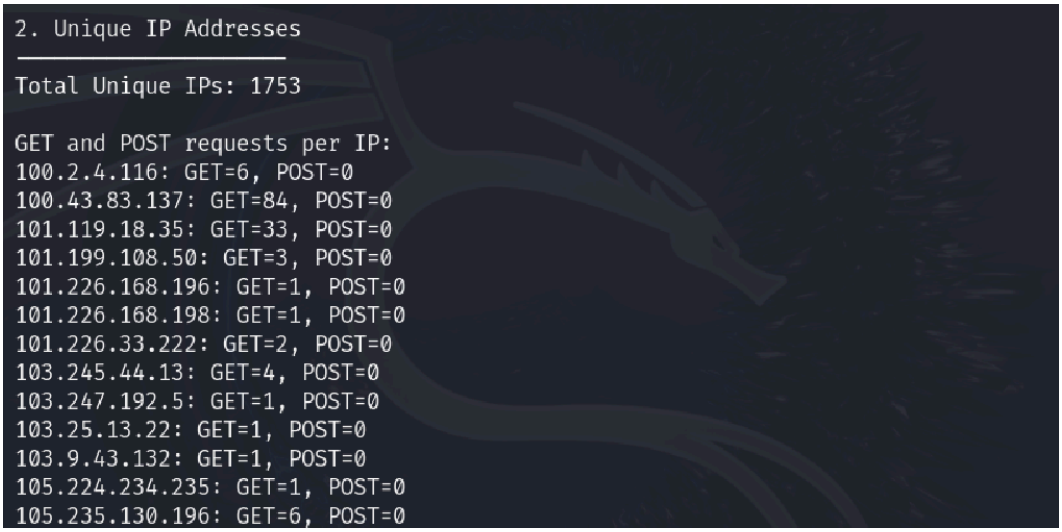
By Yehia Ahmed 2205126

Overview

This report analyzes Apache server logs to identify request patterns, failure trends, and potential security concerns. Based on the log analysis findings, the following recommendations are proposed to reduce failure rates, optimize system performance, manage peak traffic periods, and mitigate potential security threats.

1. Request Counts

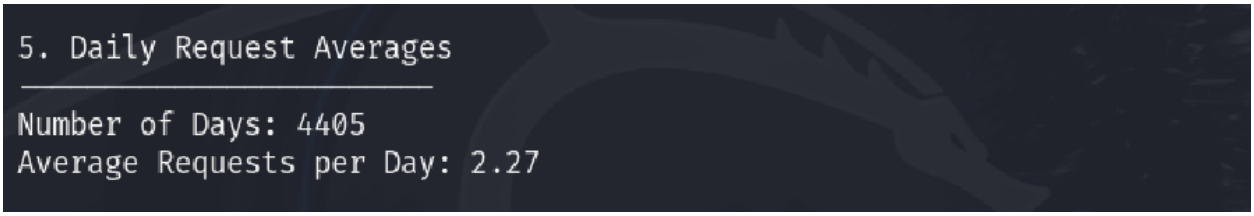
2. Unique IP Addresses



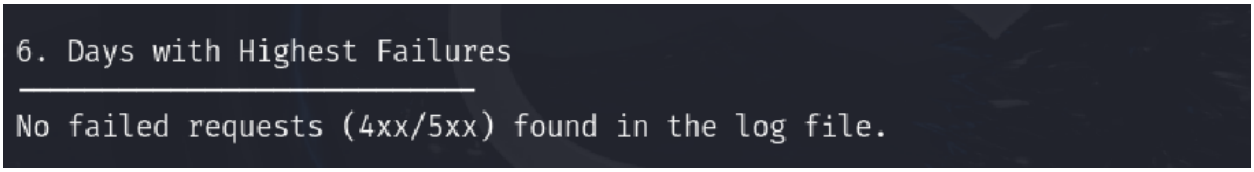
3. Failure Requests (4xx/5xx)

4. Top User

5. Daily Request Averages

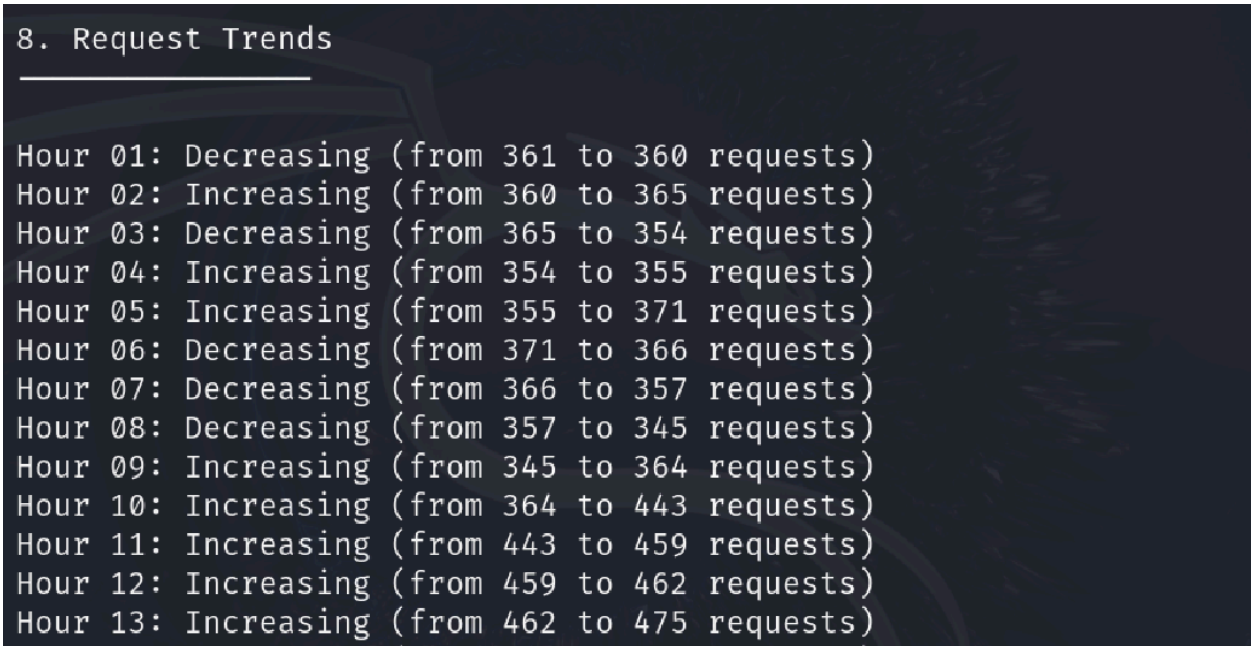


6. Days with Highest Failures

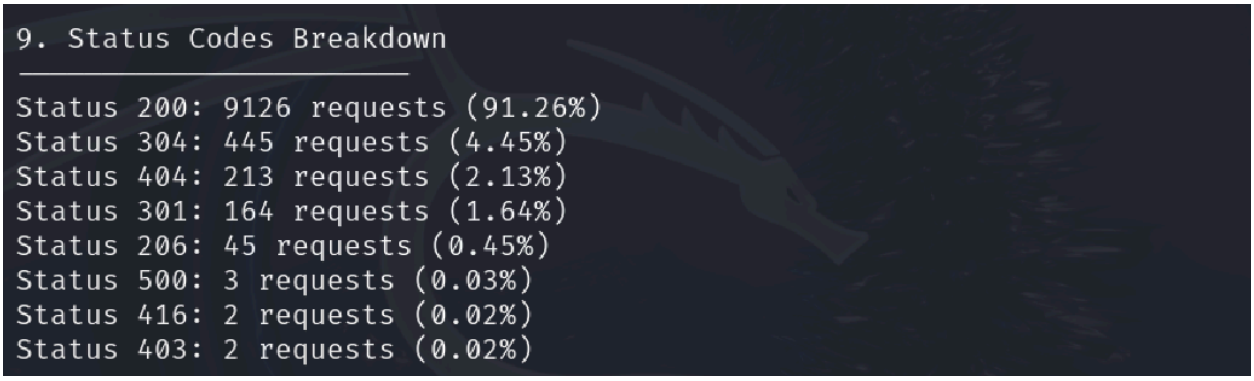


7. Requests by Hour

8. Request Trends



9. Status Codes Breakdown



10. Most Active User by Method & 11. Patterns in Failure Requests

12. Analysis Suggestions

Based on the analysis, here are some observations and recommendations:

##1. Reducing Failures

- **Address 404 Errors (213 occurrences):**

 - Implement redirects for common 404 errors or update broken links on the website.

- Analyze the requested URLs causing 404 errors to identify misconfigured resources or outdated content.

- **Mitigate 500 Errors (3 occurrences):**

- Investigate server-side issues, such as application bugs or resource exhaustion, that may cause internal server errors.
- Enable detailed error logging to pinpoint the root causes of 500 errors.

- **Handle 403 and 416 Errors:**

- Review access controls for 403 errors to ensure legitimate users are not being blocked.
- For 416 errors (range requests), verify that the server correctly handles partial content requests, especially for large files.

- **Proactive Monitoring:**

- Set up real-time alerts for spikes in 4xx/5xx errors to address issues promptly.
- Use log analysis tools (e.g., ELK Stack, Splunk) to track failure patterns and correlate them with server performance metrics.

2. Addressing High-Traffic Days and Times

- **Peak Hours (14:00–15:00):**

- Scale server resources (e.g., CPU, memory, or load balancers) during 14:00–15:00 to handle peak traffic (approximately 498 requests/hour).
- Implement caching for static content (e.g., images, CSS, JavaScript) to reduce server load during these hours.

- **High-Failure Days (May 18–20, 2015):**

- Investigate server logs for May 18–20, 2015, to identify specific events (e.g., maintenance, updates, or attacks) that may have caused increased failures.
- Schedule maintenance or updates outside peak traffic days to minimize disruptions.

- **Hourly Failure Patterns:**

- Focus on 09:00, 05:00, and 06:00, which show elevated failure rates (18, 15, and 14 failures, respectively).
- Correlate these failures with server load, backups, or scheduled tasks to identify potential causes.

3. Security Concerns and Anomalies

- **High Request Volume from Single IPs:**

- **66.249.73.135 (482 requests)**: Likely a search engine crawler (e.g., Googlebot). Verify its legitimacy using reverse DNS lookup or user-agent analysis. If legitimate, ensure the server is optimized for crawler traffic (e.g., sitemaps, robots.txt).
- **46.105.14.53 (364 requests) and 130.237.218.86 (357 requests)**: Investigate these IPs for potential scraping or malicious activity. Check user-agent strings and request patterns (e.g., rapid requests, unusual URLs).
- Implement rate-limiting for IPs exceeding a threshold (e.g., 300 requests/hour) to prevent abuse while allowing legitimate crawler activity.

- **Unusual POST Activity:**

- Investigate **78.173.140.106 (3 POST requests)**, as POST requests are rare in this dataset. Check the requested endpoints and payloads for signs of unauthorized access or injection attempts.
- Strengthen input validation and CSRF protections for POST endpoints to prevent exploits.

- **General Security Measures:**

  - Deploy a Web Application Firewall (WAF) to detect and block suspicious traffic patterns.
  - Monitor for rapid, repeated requests from the same IP within short time spans, which could indicate brute-force attacks or scraping.
  - Regularly update server software and apply security patches to mitigate vulnerabilities.

4. System and Service Improvements

- **Caching and Content Delivery:**

  - Implement a Content Delivery Network (CDN) to offload static content and reduce server load, especially for high-traffic IPs like crawlers.
  - Use server-side caching (e.g., Redis, Memcached) for frequently accessed dynamic content to improve response times.

- **Load Balancing:**

  - Deploy load balancers to distribute traffic across multiple servers, particularly during peak hours (14:00–15:00).
  - Use auto-scaling groups in cloud environments to dynamically adjust resources based on traffic.

Conclusion

Apache server logs indicate a stable system with a low failure rate (2%), yet there are clear opportunities to enhance performance and security. Addressing 404 errors, scaling resources during peak hours, investigating high-failure days, and monitoring high-traffic IPs can significantly improve system reliability and efficiency. Implementing caching mechanisms, load balancing, and security

measures such as rate limiting and Web Application Firewalls (WAFs) will further strengthen the service. Regular monitoring and automated log analysis are essential for the proactive detection and resolution of issues.