Information Security

Doctor: Mohamed Hassan

Instructor: Yahia Ashraf

Name: **Bassel Yasser Abdelsamea**
ID: 2205015

Apache Logs Analysis Report

Contents

1 Overview

This report analyzes the Apache server logs to identify request patterns, failure trends, and potential security concerns. Based on the provided log analysis output, the following suggestions aim to reduce failures, optimize performance, address high-traffic periods, and mitigate potential security risks.

2 Analysis Findings

2.1 Request Volume and Patterns

- **Total Requests**: 10,000, with 9,952 GET and 5 POST requests.
- **Peak Request Hour**: 14:00 (498 requests), followed by 15:00 (496 requests).
- **Average Requests per Day**: 2,500.
- **Most Active IP**: 66.249.73.135 (482 GET requests).

2.2 Failure Trends

- **Failed Requests (4xx/5xx)**: 220 (2% of total requests).
- **Highest Failure Days**: May 19, 2015 (66 failures), May 18, 2015 (66 failures), May 20, 2015 (58 failures).
- **Peak Failure Hour**: 09:00 (18 failures), followed by 05:00 (15 failures).
- **Status Code Breakdown**: 404 (213), 403 (2), 416 (2), 500 (3).

2.3 IP Activity

- **Total Unique IPs**: 1,753.
- **Notable IPs with High Activity**:
  - 66.249.73.135: 482 GET requests (likely a crawler, e.g., Googlebot).
  - 46.105.14.53: 364 GET requests.
  - 130.237.218.86: 357 GET requests.
  - 75.97.9.59: 273 GET requests.

2.4 Security Observations

- The high request volume from 66.249.73.135 suggests automated bot or crawler activity.
- Limited POST requests (5 total) reduce the likelihood of form-based attacks, but one IP (78.173.140.106) made 3 POST requests, which is unusual given the low overall POST activity.
- No immediate evidence of distributed denial-of-service (DDoS) attacks, but high request volumes from specific IPs warrant monitoring.

3 Suggestions for Improvement

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  - Analyze the requested URLs causing 404 errors to identify misconfigured resources or outdated content.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

3.2 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.

3.3 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.

3.4 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.
- **Log Analysis and Monitoring**:

  - Automate log analysis with tools like Logstash or Graylog to detect anomalies in real time.
  - Set up dashboards to visualize request patterns, failure rates, and IP activity for easier monitoring.

- **Crawler Optimization**:
  - Optimize the website for search engine crawlers by providing a clear sitemap and using robots.txt to control access to sensitive areas.
  - Monitor crawler activity to ensure it does not overwhelm server resources.

- **Performance Testing**:
  - Conduct stress tests to determine the servers capacity during peak traffic and identify bottlenecks.
  - Optimize database queries and server configurations to handle high request volumes efficiently.

4 Conclusion

The Apache server logs reveal a stable system with a low failure rate (2%), but there are opportunities to enhance performance and security. By addressing 404 errors, scaling resources during peak hours, investigating high-failure days, and monitoring high-traffic IPs, the system can achieve greater reliability and efficiency. Implementing caching, load balancing, and security measures like rate-limiting and WAFs will further improve the service. Regular monitoring and automated log analysis will ensure proactive identification and resolution of issues.