Log File Analysis With Bash Script

Name : Mayssoune hussein ElMasry

ID:2205251

Information Security Management

Eng/Yehia Ashraf

CONTENTS

01

INTRODUCTION

Executive Summary:

This report presents a comprehensive analysis of the web server log file access.log using a Bash script to extract key statistics and insights. The analysis covers request counts, unique IP addresses, failure rates, top users, daily averages, failure patterns, hourly requests, request trends, status code breakdowns, and active users by method. Key findings include a high proportion of GET requests, a single failure (404 status code), and concentrated request activity from one IP address. Recommendations include implementing rate limiting, investigating the 404 error, and optimizing server performance for peak hours.

Introduction:

The objective of this analysis is to examine the web server log file access.log to understand request patterns, identify potential issues, and suggest improvements. The log file, spanning from May 17 to May 20, 2015, contains entries in the Apache combined log format. A Bash script was developed to process the log file and generate statistics for 11 specified requirements. This report details the findings and provides

02

ANALYSIS RESULTS

The Analysis Results section presents the findings from processing the web server log file access.log using the Bash script log_analysis.sh. The log file, containing 125 entries from May 17 to May 20, 2015, follows the Apache combined log format, capturing details such as IP addresses, timestamps, request methods, URLs, status codes, and user agents. The script was designed to address 11 specific requirements, including request counts, unique IP addresses, failure rates, top users, daily averages, and temporal patterns in requests and failures. Each requirement was analyzed using standard Unix tools (e.g., awk, grep, sort, uniq) to extract and summarize relevant metrics. The results provide insights into server usage patterns, performance, and potential issues, forming the basis for actionable recommendations. The following subsections detail the findings for each requirement, supported by quantitative metrics and qualitative observations, with outputs stored in timestamped directories for reference.

1 Request Counts

• Total requests: 125

• GET requests: 125

• POST requests: 0 All requests were GET requests, indicating the server primarily serves static content or read-only operations.

2 Unique IP Addresses

• Total unique IPs: 36

• Detailed counts: Each IP made only GET requests, with no POST requests observed.

For example: – 83.149.9.216: 23 GET requests – 66.249.73.135: 13

3 Failure Requests

• Failed requests: 1 (404 status code)

• Failure percentage: 0.80% The low failure rate indicates robust server

4 Top User

• Most active IP: 83.149.9.216 (23 requests) This IP’s high activity warrants further investigation for potential abuse or legitimate heavy

5 Daily Request Averages

• Unique days: 4 (May 17–20, 2015)

• Average requests per day: 31.25 The consistent daily request volume

6 Failure Analysis

• Days with failures: Only one failure on May 17, 2015 The single failure does not indicate a recurring issue but requires attention to prevent

7 Requests By Hour

Requests were distributed across hours, with peak activity at:

• 10:00: 80 requests

• 11:00: 25 requests

• 20:00: 10 requests

• 21:00: 10 requests The 10:00 hour saw significantly higher traffic, likely due to crawler activity.

8 Requests By Trends

• From 10:00 to 11:00: Decreased by 55 requests

• From 11:00 to 20:00: Decreased by 15 requests

• From 20:00 to 21:00: Remained stable The sharp drop after 10:00 suggests a burst of activity followed by normal usage.

9 Status Code Breakdown

• 200: 114 requests

• 304: 6 requests

• 404: 1 request The dominance of 200 status codes confirms successful request handling, with 304 indicating effective caching.

10 Most Active User By Method

• Most active GET user: 83.149.9.216 (23 GET requests)

• Most active POST user: None (0 POST requests) The absence of POST requests aligns with the server’s read-only usage pattern.

11 Patterns in Faiure Requests

• Failure by hour: Single 404 at 10:00 on May 17, 2015

• Failure by day: Single failure on May 17, 2015 No recurring failure patterns were observed, but the 404 error should be investigated.

03

Suggestions For Improvement

Failure Reduction:

• Investigate the 404 error for /doc/index.html?... to determine if it was a user error or a misconfigured link. Ensure all resources are accessible or redirect to valid pages.
• Implement monitoring to alert administrators of future 4xx/5xx errors.

Resource Allocation:

• Optimize server capacity for the 10:00 hour, which sees peak traffic (80 requests). Consider load balancing or caching enhancements.
• Schedule maintenance during low-traffic hours (e.g., 20:00–21:00) to minimize disruption.

IMPROVEMENTS

Security Concerns:

• Monitor IP 83.149.9.216 for potential abuse, as it accounts for 18.4% of requests. Implement rate limiting to prevent excessive requests from single IPs.
• Review crawler activity (e.g., Googlebot, Sogou) to ensure compliance with robots.txt

System Improvements:

• Enhance caching strategies, as 304 responses indicate effective cache usage. Expand caching for frequently accessed resources.
• Implement detailed logging for POST requests if they are expected in the future, to enable similar analysis.
• Deploy intrusion detection systems to flag rapid successive

04

Conclusion

The analysis of access.log reveals a stable web server with a low failure rate and consistent daily request volumes. The dominance of GET requests and high activity from a single IP (83.149.9.216) suggest a read-only usage pattern with potential crawler activity. The single 404 error requires investigation, and peak traffic at 10:00 necessitates resource optimization. Implementing the proposed recommendations will enhance server reliability, security, and performance. Future analyses should monitor for POST requests and recurring failures to maintain system health.

Appendix:

Bash Script

The analysis was performed using the Bash script log_analysis.sh, available at https://github.com/mayssouneelmasry/log-analysis.

Key components include:

• Request counting with grep and wc

• IP analysis with cut, sort, and uniq

• Failure detection with awk

Thank you for watching!