Log File Analysis Report

Student Name: Kareem walid saad

Id: 2205076

1. Objective

The objective of this task is to analyze a web server log file using a Bash script to extract useful statistics, identify anomalies, detect failure patterns, and suggest system improvements based on the request behavior.

2. Summary of Features Implemented in the Script

- Total request count
- Request method counts (GET, POST, PUT, DELETE)
- Unique IP address count
- Top 10 most active IP addresses
- Most active IP overall
- GET and POST request count per IP address
- Total number of failed requests (HTTP 4xx/5xx)
- Failure percentage relative to total requests
- Most frequent HTTP status code
- Hourly distribution of requests
- Hourly distribution of failed requests
- Days with highest failure counts
- Average number of requests per day
- Busiest hour of the day (highest traffic)

3. Results from Sample Log File

- Total Requests: 4
- GET Requests: 2
- POST Requests: 2
- PUT Requests: 0
- DELETE Requests: 0
- Unique IPs: 3
- Failed Requests: 2 (401 and 500)
- Failure Rate: 50%
- Most Active IP: 192.168.1.1
- Most Common Status Code: 200
- Daily Average Requests: 4.0

- Busiest Hour: 10:00 (2 requests)

4. Analysis & Observations

- The failure rate is significantly high (50%), indicating the presence of authentication or server-side issues.
- IP 192.168.1.1 is the most active, which may require monitoring in production environments.
- The presence of HTTP 401 errors indicates unauthorized access attempts, possibly brute-force login attacks.
- HTTP 500 internal server error suggests backend issues that must be debugged and fixed.
- Request traffic was light and only spanned over one day, with hour 10 having the most requests.

5. Recommendations

- Implement rate limiting and IP banning mechanisms to prevent abuse by overly active IPs.
- Add authentication hardening features such as CAPTCHA or account lockouts to reduce 401 errors.
- Investigate server logs and backend code to determine the cause of HTTP 500 responses.
- Monitor daily traffic patterns to prepare for scaling or load balancing.
- Expand logging to include additional headers like User-Agent, Referer, and response times.

6. Conclusion

The Bash-based log analysis tool provides a practical and efficient way to gain insights into server behavior, usage trends, and system errors.
It is essential for identifying problems early, maintaining service reliability, and enhancing system security.
This analysis provides a solid foundation for operational monitoring and incident response.

7. Script Design & Implementation Notes

The Bash script was designed using modular and efficient shell commands to extract key metrics from standard web server log files.
Key design principles included:

- Using AWK for pattern matching and field extraction due to its speed and line-by-line processing.
- Grep and sort were used for quick counts and filtering specific request methods.
- Date-time fields were parsed for time-based analysis such as hourly traffic and daily averages.
- All values were calculated dynamically from the log file, making the script reusable and scalable.
- Output formatting uses printf and echo with headers for easy reading.

The script supports common HTTP methods (GET, POST, PUT, DELETE) and handles various HTTP status codes.
The logic was tested against a sample log file and can scale for larger datasets.

8. Future Enhancements & Automation

To further improve this system and expand its use in production, the following features are recommended:

1. Automate script execution with CRON jobs to generate daily/weekly reports.
2. Export analysis results to CSV or JSON format for integration with dashboards (e.g., Grafana).
3. Send alerts (email, Slack) if error rate exceeds thresholds.
4. Integrate with a centralized log collector (like ELK Stack or Graylog).
5. Include response time monitoring and flag slow responses.
6. Add support for analyzing logs from multiple servers.

These improvements will transform the current script into a robust part of a full observability and incident response solution.