

Gateway Firewall Log Analysis and Security Reporting System

Project Report

Submitted for Academic Internship

Submitted by

Nirjara Jain

Final Year Computer Engineering Student
Gujarat Technological University (GTU)

Submitted to

[Organization / Institute Name]

December 2025

Contents

| | | |
|-----------|---------------------------------------|-----------|
| 1 | Introduction | 2 |
| 2 | Objectives | 3 |
| 3 | Scope of the Project | 4 |
| 4 | Technologies Used | 5 |
| 5 | System Architecture | 6 |
| 6 | Project Structure | 7 |
| 7 | Implementation Details | 8 |
| 7.1 | Log File Processing | 8 |
| 7.2 | Data Extraction | 8 |
| 7.3 | Report Generation | 8 |
| 8 | Sample Output | 9 |
| 9 | Use Cases | 10 |
| 10 | Limitations | 11 |
| 11 | Future Enhancements | 12 |
| 12 | Conclusion | 13 |
| 13 | References | 14 |
| 14 | Screenshots and Results | 15 |
| 14.1 | Project Directory Structure | 15 |
| 14.2 | Firewall Log Sample | 15 |
| 14.3 | Script Execution | 16 |
| 14.4 | Generated Output Report | 16 |

1. Introduction

Modern organizations face continuous cyber threats targeting their network infrastructure. Firewalls act as the first line of defense by filtering malicious traffic and enforcing security policies. However, firewall devices generate a large volume of logs, making manual analysis inefficient and error-prone.

This project focuses on designing and implementing a Gateway Firewall Log Analysis system that processes firewall logs, extracts meaningful security information, and generates structured reports. The system assists security teams in understanding attack patterns and improving network security posture.

2. Objectives

The objectives of this project are:

- To analyze gateway-level firewall logs efficiently
- To extract blocked and denied traffic information
- To identify source and destination IP address patterns
- To generate time-based security reports
- To assist administrators in proactive security decisions

3. Scope of the Project

The scope of this project includes parsing firewall log files, extracting key security-related fields, and generating structured reports. The system focuses only on log analysis and reporting and does not involve active firewall rule configuration or enforcement.

4. Technologies Used

- Operating System: Linux (Kali Linux)
- Programming Language: Python 3
- Tools: Nano Editor
- File Formats: .log, .csv

5. System Architecture

The system follows a modular workflow:

1. Firewall generates gateway-level logs
2. Logs are stored locally
3. Python script parses and processes logs
4. Extracted data is structured into CSV format
5. Reports are generated for analysis

6. Project Structure

```
gateway-firewall-log-analyzer/  
  logs/  
    firewall.log  
  scripts/  
    analyze_logs.py  
  output/  
    firewall_report.csv  
  report/  
    Project_Report.pdf  
  README.md
```


7. Implementation Details

7.1 Log File Processing

The firewall log file is read line by line using Python. Each entry is parsed to extract timestamp, action, source IP, and destination IP.

7.2 Data Extraction

Only relevant security actions such as BLOCK and DENY are processed for analysis.

7.3 Report Generation

The processed data is written into a CSV file for easy analysis and reporting.

8. Sample Output

The generated report includes:

- Date
- Time
- Action
- Source IP Address
- Destination IP Address

9. Use Cases

- Monitoring blocked inbound and outbound traffic
- Identifying repeated malicious IP addresses
- Supporting security audits
- Assisting SOC analysis workflows

10. Limitations

- Static log analysis only
- No real-time monitoring
- No visualization dashboards

11. Future Enhancements

- Integration with SIEM platforms
- Real-time syslog ingestion
- Visualization dashboards
- Automated alerting mechanisms

12. Conclusion

This project demonstrates an effective approach to gateway firewall log analysis using Python. It enables structured reporting and security insights that can assist administrators in proactive defense strategies. The project provides a solid foundation for further expansion into SIEM-based monitoring.

13. References

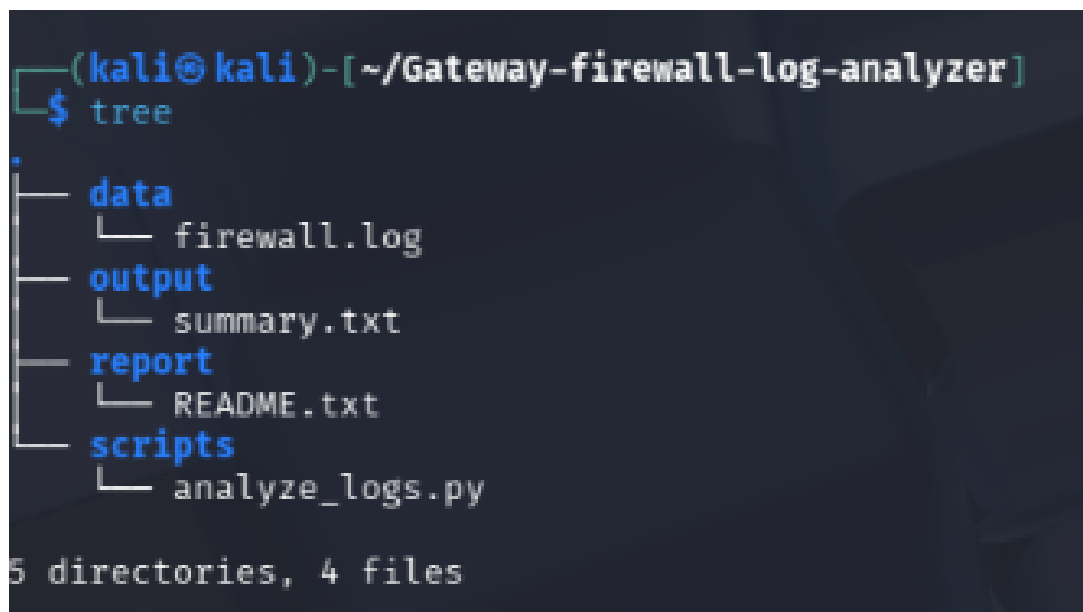
- Firewall Log Analysis Concepts
- Python Documentation
- Network Security Fundamentals

14. Screenshots and Results

This section presents important screenshots captured during the development and execution of the project. These figures demonstrate the project workflow, script execution, and generated outputs.

14.1 Project Directory Structure

Figure 14.1 shows the organized directory structure of the Gateway Firewall Log Analyzer project.

A terminal window screenshot showing the output of the 'tree' command in a Kali Linux environment. The prompt is '(kali@kali)-[~/Gateway-firewall-log-analyzer]'. The command '\$ tree' has been executed, resulting in a tree-like structure of the project directory. The structure shows four subdirectories: 'data', 'output', 'report', and 'scripts'. The 'data' directory contains 'firewall.log'. The 'output' directory contains 'summary.txt'. The 'report' directory contains 'README.txt'. The 'scripts' directory contains 'analyze_logs.py'. At the bottom, it states '5 directories, 4 files'.

```
(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ tree
.
├── data
│   └── firewall.log
├── output
│   └── summary.txt
├── report
│   └── README.txt
└── scripts
    └── analyze_logs.py

5 directories, 4 files
```

Figure 14.1: Project directory structure of the Gateway Firewall Log Analyzer

14.2 Firewall Log Sample

Figure 14.2 shows a sample of the firewall log file used as input for analysis. The log contains information such as timestamp, action, source IP, destination IP, protocol, and port.


```

(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ cd data

(kali@kali)-[~/Gateway-firewall-log-analyzer/data]
$ ls
firewall.log

(kali@kali)-[~/Gateway-firewall-log-analyzer/data]
$ head firewall.log
Dec 20 10:15:21 firewall kernel: INBOUND BLOCK SRC=192.168.1.45 DST=10.0.0.5 PROTO=TCP DPT=22
Dec 20 10:17:45 firewall kernel: OUTBOUND BLOCK SRC=10.0.0.5 DST=185.220.101.12 PROTO=TCP DPT=80
Dec 20 11:05:10 firewall kernel: INBOUND BLOCK SRC=203.0.113.78 DST=10.0.0.5 PROTO=UDP DPT=53
Dec 21 09:55:32 firewall kernel: INBOUND BLOCK SRC=45.83.64.22 DST=10.0.0.5 PROTO=TCP DPT=443
Dec 21 14:12:09 firewall kernel: OUTBOUND BLOCK SRC=10.0.0.5 DST=91.121.83.100 PROTO=TCP DPT=8080

```

Figure 14.2: Sample firewall log file used for analysis

14.3 Script Execution

Figure 14.3 shows the execution of the Python script used to parse and analyze the firewall logs.

```

(kali@kali)-[~]
$ cd Gateway-firewall-log-analyzer

(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ mkdir logs scripts output report

(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ nano logs/firewall.log

(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ nano scripts/analyze_logs.py

(kali@kali)-[~/Gateway-firewall-log-analyzer]
$ cd scripts

(kali@kali)-[~/Gateway-firewall-log-analyzer/scripts]
$ python3 analyze_logs.py

Firewall log analysis completed.

```

Figure 14.3: Execution of firewall log analysis script

14.4 Generated Output Report

Figure 14.4 shows the generated CSV report containing summarized firewall statistics such as blocked IPs and timestamps.

```
(kali㉿kali)-[~/Gateway-firewall-log-analyzer/scripts]
$ python3 analyze_logs.py

Firewall log analysis completed.

(kali㉿kali)-[~/Gateway-firewall-log-analyzer/scripts]
$ cat ../output/summary.txt

Dec 20 | BLOCK | 3 blocked connections
Dec 21 | BLOCK | 2 blocked connections

(kali㉿kali)-[~/Gateway-firewall-log-analyzer/scripts]
$ nano report/README.txt
```

Figure 14.4: Generated firewall analysis report