Abhishek Deshpande (z5419626) Tutor: Halliya Um

COMP6441 Project Report

GitHub Repository: https://github.com/21-abhi-d/comp6441_project
Project Description: Log-Based Threat Detection & Explanation Tool
This project aimed to build a lightweight, end-to-end system that detects, explains, and contextualizes security threats hidden in server log files. By combining custom rule-based detection with an embedded LLM (Large Language Model) interface, the tool not only identifies suspicious activity, like brute-force login attempts or malicious web access, but also translates those findings into understandable, actionable insights for users with limited security expertise.

Why This Is Important:
Log files are rich with signals of cyberattacks, but most small businesses and solo developers either:

1. Lack the expertise to interpret raw logs, or
2. Can't afford enterprise SIEM software that automates threat detection.

This results in serious blind spots, where threats go unnoticed until damage is done.

Results:
The final outcome of this project is a fully functional log-based threat detection system that automatically parses, analyzes, and explains security-relevant events from system logs. The system successfully detects brute-force attacks, suspicious HTTP requests, and other anomalies across both auth.log and access.log files.
The tool features:

- A modular pipeline consisting of custom parsers, detectors, formatters, and exporters.
  A built-in LLM-powered explanation engine that provides plain-English summaries and Q&A for detected anomalies, enabling better understanding for non-experts.
- Support for structured output formats (CSV/JSON) to facilitate archiving or deeper external analysis.

Throughout development, extensive testing was performed on real and synthetic log files to validate detection accuracy and explanation quality. Detection outputs were cross-checked for false positives and ambiguous events were fed back into the explanation system for iterative improvement.

Screenshots in the appendix show:

- Sample log input lines *(Ref. 1)*
- Detection outputs for brute-force and web attack patterns *(Ref. 2)*
- LLM-generated summaries contextualising the events *(Ref. 3)*
- Architecture and flow diagrams of the full pipeline *(Ref. 4)*

What I'm most proud of is the integration of technical depth with accessibility. This isn't just a detection tool, but one that empowers small teams and individual users to understand their security posture, without relying on expensive or complex software.

Challenges faced and how I resolved them

Challenge	Description	Resolutive Action	Outcome
Inconsistent Log Formats	Log parsing failed due to inconsistent spacing, missing fields, or varied timestamp formats across auth.log and access.log.	Developed regex with fallback conditions and structured error handling in the parser.	Parser now handles malformed or inconsistent entries robustly.
LLM Generating Vague Summaries (hallucination)	Initial outputs lacked insight, restating logs without meaningful explanation.	Refined prompt design by injecting structured metadata (e.g. IP, event type, timestamp) into context.	LLM now generates clearer, contextual summaries.
Time & Architecture Management	Code was initially too linear and difficult to scale or test.	Refactored into a modular pipeline: Parser → Detectors → Formatter → Exporter → LLM. Used weekly planning to manage time.	Cleaner architecture, better scalability, and easier debugging.

Reflection:

Working on this project deepened my understanding of how foundational yet underutilised system logs are in the broader context of cybersecurity. In an era where attacks are becoming more sophisticated but also more automated and opportunistic, it was eye-opening to realise that many breaches, especially brute-force attacks or web enumeration, leave clear traces in logs long before any real damage is done. Yet, because these logs are messy and unstructured, they're often ignored by those who need them most: small businesses, students managing their own servers, or understaffed tech teams.

Through building this tool, I became more aware of the gap between raw technical visibility and human understanding. The detectors alone were not enough, it was the integration of a context-aware LLM that made the system truly useful. I learned that cybersecurity is not just about catching threats, but also about empowering people to understand and act on what they see. This project taught me the value of designing not just for function, but for accessibility and clarity. It's a small step toward making security more inclusive, and that's something I'll carry into future work.

In terms of strengths, I'm particularly proud of the modular architecture and the thoughtful integration of explainability through LangChain and OpenAI. The pipeline was designed with extensibility in mind, and the end-user interaction loop added a meaningful layer of usability that many traditional detection tools lack. I also made a conscious effort to test the system against both clean and noisy log data, which helped surface edge cases early in development.

However, this project also had its weaknesses. One limitation was the dependency on handcrafted rules for detection, while effective for known patterns, they may struggle with novel attack vectors or obfuscated behavior. Additionally, tuning the LLM outputs was a non-trivial task; early responses were vague or generic, and improving prompt structure took several iterations. Finally, my time was stretched between coding and documentation, and I would have liked to spend more time exploring real-time detection or alerting features.

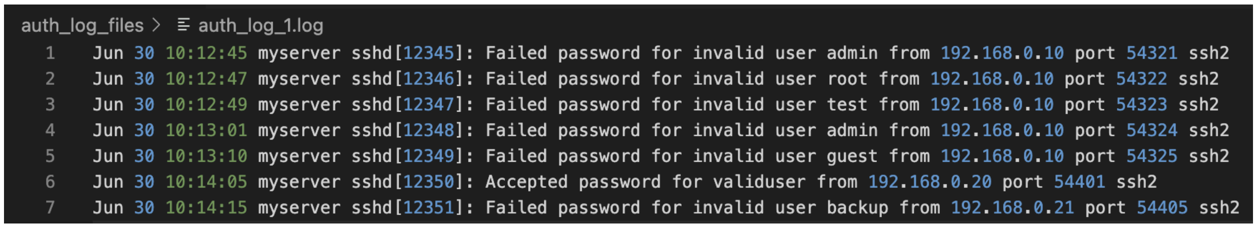
Still, these challenges were deeply formative. They pushed me to think critically about user needs, to iterate quickly, and to seek out clarity over complexity. Ultimately, this project changed the way I think about cybersecurity, not just as a technical discipline, but as a space where design, communication, and accessibility matter just as much.

References

1. https://www.verizon.com/business/resources/reports/dbir/
2. https://www.ponemon.org/library
3. https://httpd.apache.org/docs/current/logs.html
4. https://platform.openai.com/docs
5. https://owasp.org/www-community/attacks/Brute_force_attack

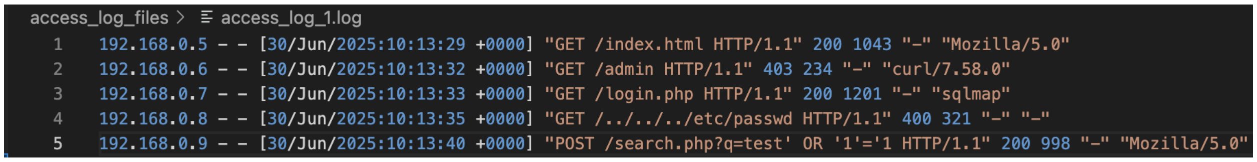
Appendix

Reference 1 - Sample log input lines



auth_log_files > auth_log_1.log

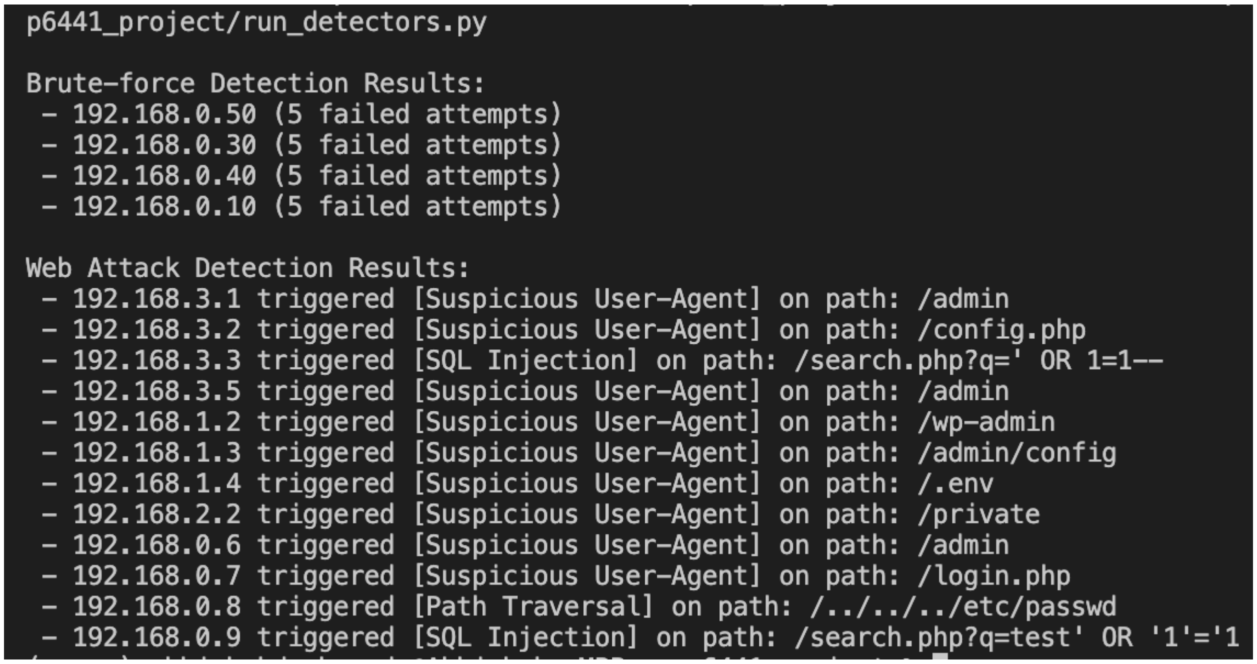
```
Jun 30 10:12:45 myserver sshd[12345]: Failed password for invalid user admin from 192.168.0.10 port 54321 ssh2
Jun 30 10:12:47 myserver sshd[12346]: Failed password for invalid user root from 192.168.0.10 port 54322 ssh2
Jun 30 10:12:49 myserver sshd[12347]: Failed password for invalid user test from 192.168.0.10 port 54323 ssh2
Jun 30 10:13:01 myserver sshd[12348]: Failed password for invalid user admin from 192.168.0.10 port 54324 ssh2
Jun 30 10:13:10 myserver sshd[12349]: Failed password for invalid user guest from 192.168.0.10 port 54325 ssh2
Jun 30 10:14:05 myserver sshd[12350]: Accepted password for validuser from 192.168.0.20 port 54401 ssh2
Jun 30 10:14:15 myserver sshd[12351]: Failed password for invalid user backup from 192.168.0.21 port 54405 ssh2
```



access_log_files > access_log_1.log

```
192.168.0.5 - - [30/Jun/2025:10:13:29 +0000] "GET /index.html HTTP/1.1" 200 1043 "-" "Mozilla/5.0"
192.168.0.6 - - [30/Jun/2025:10:13:32 +0000] "GET /admin HTTP/1.1" 403 234 "-" "curl/7.58.0"
192.168.0.7 - - [30/Jun/2025:10:13:33 +0000] "GET /login.php HTTP/1.1" 200 1201 "-" "sqlmap"
192.168.0.8 - - [30/Jun/2025:10:13:35 +0000] "GET /../../../etc/passwd HTTP/1.1" 400 321 "-" "-"
192.168.0.9 - - [30/Jun/2025:10:13:40 +0000] "POST /search.php?q=test' OR '1'='1 HTTP/1.1" 200 998 "-" "Mozilla/5.0"
```

Reference 2 - Detector Outputs for brute force and web-attack detectors



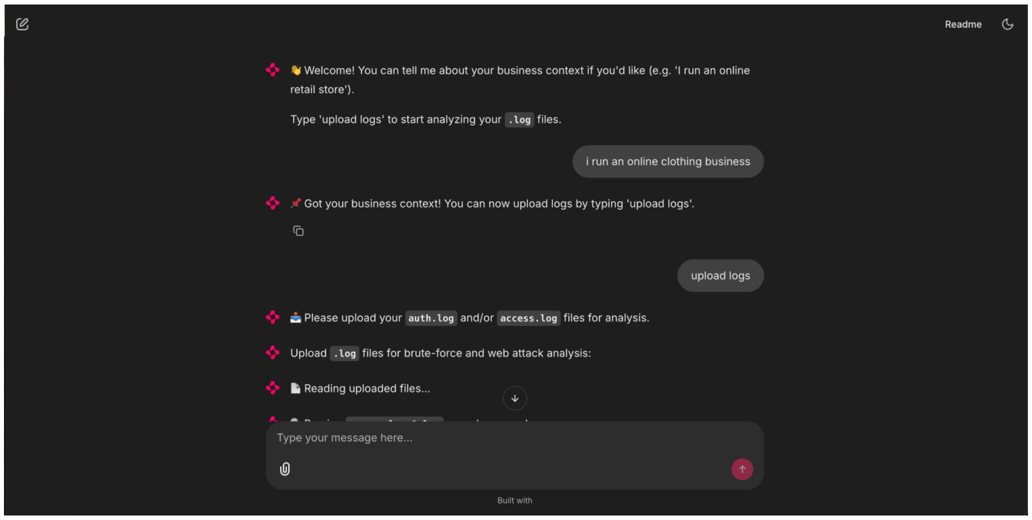
```
p6441_project/run_detectors.py

Brute-force Detection Results:
 - 192.168.0.50 (5 failed attempts)
 - 192.168.0.30 (5 failed attempts)
 - 192.168.0.40 (5 failed attempts)
 - 192.168.0.10 (5 failed attempts)

Web Attack Detection Results:
 - 192.168.3.1 triggered [Suspicious User-Agent] on path: /admin
 - 192.168.3.2 triggered [Suspicious User-Agent] on path: /config.php
 - 192.168.3.3 triggered [SQL Injection] on path: /search.php?q=' OR 1=1--
 - 192.168.3.5 triggered [Suspicious User-Agent] on path: /admin
 - 192.168.1.2 triggered [Suspicious User-Agent] on path: /wp-admin
 - 192.168.1.3 triggered [Suspicious User-Agent] on path: /admin/config
 - 192.168.1.4 triggered [Suspicious User-Agent] on path: /.env
 - 192.168.2.2 triggered [Suspicious User-Agent] on path: /private
 - 192.168.0.6 triggered [Suspicious User-Agent] on path: /admin
 - 192.168.0.7 triggered [Suspicious User-Agent] on path: /login.php
 - 192.168.0.8 triggered [Path Traversal] on path: /../../../etc/passwd
 - 192.168.0.9 triggered [SQL Injection] on path: /search.php?q=test' OR '1'='1
```

Abhishek Deshpande (z5419626) Tutor: Halliya Um

Reference 3 - LLM-generated summaries contextualising the events



👋 Welcome! You can tell me about your business context if you'd like (e.g. 'I run an online retail store').

Type 'upload logs' to start analyzing your `.log` files.

i run an online clothing business

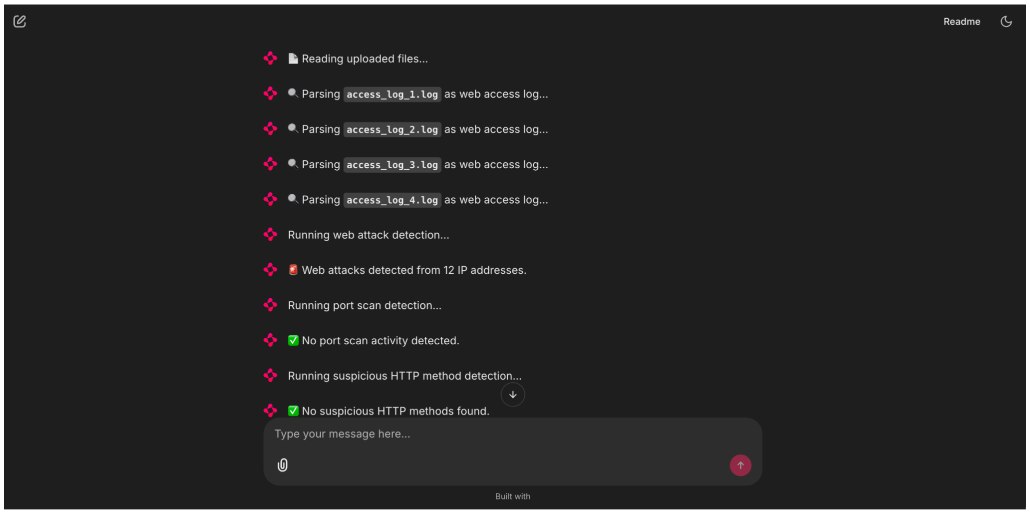
📌 Got your business context! You can now upload logs by typing 'upload logs'.

upload logs

📤 Please upload your `auth.log` and/or `access.log` files for analysis.

Upload `.log` files for brute-force and web attack analysis:

📄 Reading uploaded files...



📄 Reading uploaded files...

🔍 Parsing `access_log_1.log` as web access log...

🔍 Parsing `access_log_2.log` as web access log...

🔍 Parsing `access_log_3.log` as web access log...

🔍 Parsing `access_log_4.log` as web access log...

Running web attack detection...

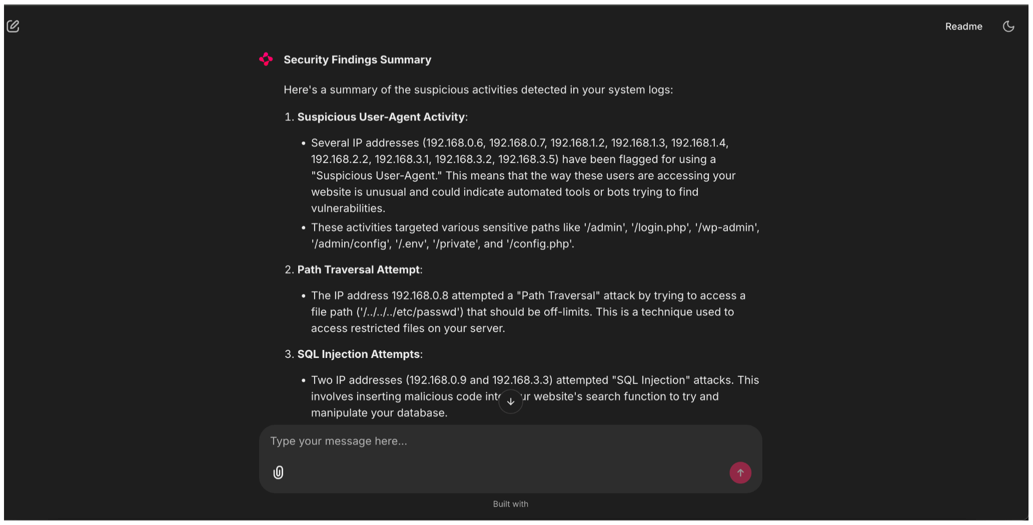
🚨 Web attacks detected from 12 IP addresses.

Running port scan detection...

✅ No port scan activity detected.

Running suspicious HTTP method detection...

✅ No suspicious HTTP methods found.



Readme

Security Findings Summary

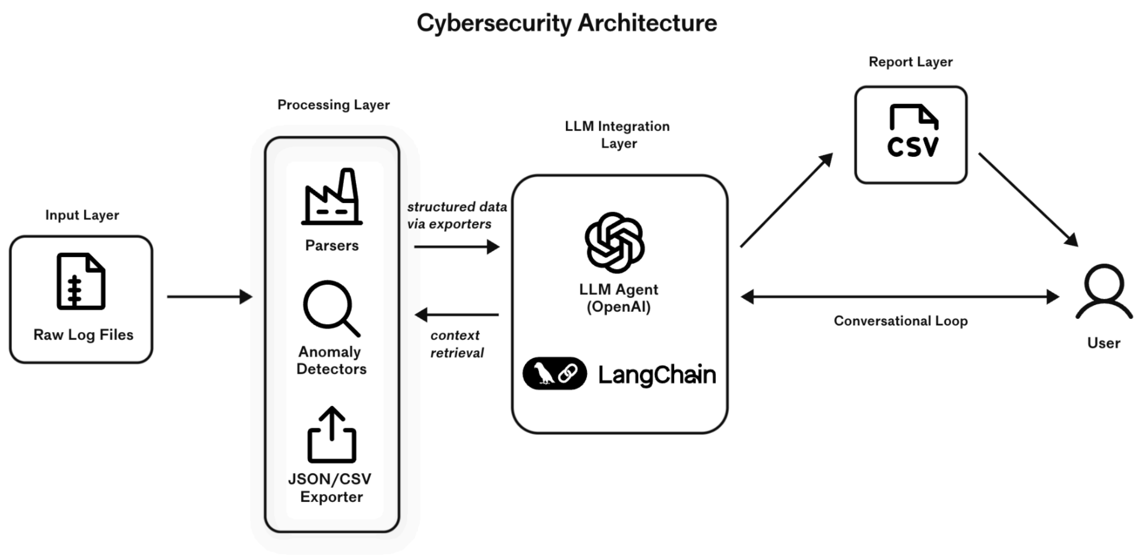
Here's a summary of the suspicious activities detected in your system logs:

1. **Suspicious User-Agent Activity**:
   - Several IP addresses (192.168.0.6, 192.168.0.7, 192.168.1.2, 192.168.1.3, 192.168.1.4, 192.168.2.2, 192.168.3.1, 192.168.3.2, 192.168.3.5) have been flagged for using a "Suspicious User-Agent." This means that the way these users are accessing your website is unusual and could indicate automated tools or bots trying to find vulnerabilities.
   - These activities targeted various sensitive paths like '/admin', '/login.php', '/wp-admin', '/admin/config', '/.env', '/private', and '/config.php'.
2. **Path Traversal Attempt**:
   - The IP address 192.168.0.8 attempted a "Path Traversal" attack by trying to access a file path ('/../../../etc/passwd') that should be off-limits. This is a technique used to access restricted files on your server.
3. **SQL Injection Attempts**:
   - Two IP addresses (192.168.0.9 and 192.168.3.3) attempted "SQL Injection" attacks. This involves inserting malicious code int[illegible]ur website's search function to try and manipulate your database.

Type your message here...

Built with

Reference 4 - Architecture Diagram


Cybersecurity Architecture
Input Layer
Raw Log Files
Processing Layer
Parsers
Anomaly Detectors
JSON/CSV Exporter
structured data via exporters
context retrieval
LLM Integration Layer
LLM Agent (OpenAI)
LangChain
Report Layer
CSV
Conversational Loop
User