Machine Learning-Based Web Log Analysis for SQL Injection Detection

Luke Kimes, Elias Alvarez, Tanay Shah
California State Polytechnic University, Pomona

INTRODUCTION

SQL injection attacks remain a significant threat to web applications, often exploited by attackers to manipulate databases and exfiltrate sensitive information. Access to web development resources is prevalent to users of all experience levels. As programing languages create more web frameworks, companies hire more web developers, and AI is utilized to write these applications, a lack of web security will become more apparent. Detecting SQLi attacks on your application can be difficult to detect and requires an understanding of how the SQL language works which many users may lack. However, machine learning (ML) "can be used to support the detection of SQL injection attacks by training a classifier to achieve the ability to recognize and therefore detect an attack" (Alghawazi, Alghazzawi, and Alarifi 765). Our mission is to utilize modern ML technology, to detect SQL injection class exploit attempts within website access logs and highlight those attempts to the user. Future implementations of the ML project has potential to be used to catch SQL injection exploits before they are able to execute. Such technology may also aid in the detection of 0-day exploits that make use of SQL injection in their exploit chain. We seek to demonstrate the feasibility of ML algorithms for modern detection, and to prove their malleability for other classes of web exploits that share similar characteristics to SQL injection.

OBJECTIVES

This project uses a machine learning-based approach to web log analysis to classify SQL queries as either benign or malicious, aimed at improving the detection of web-based threats. These SQL queries will undergo a cleaning process, where irrelevant data is removed.

Log Cleaning and Enrichment

To begin to send data to our model it first needs to be cleaned. Depending on the scenario, out project will either be fed queries from access logs or from the web application itself. Once queries are passed to us, our application will note down specifics of the query such as the presence of certain SQL operators or query characters.

Machine Learning SQL Injection Detection:

The SQL queries and their metadata will be fed into a machine learning model specifically trained to detect SQL injection attacks. The model will classify each query as either benign or malicious based on features such as query structure, keywords, and behavior patterns typically associated with SQL injections. This approach will help identify potential security threats encountered by a given application.

User Output:

Queries identified as malicious will invoke a response that indicates as such. From here an application or user may distinctly see whether or not a given query is considered to be safe. In the case of an application, execution of the malicious query could be avoided all together. In the case of a user, security professionals may be able to use this new insight to identify malicious behavior and further incident response investigations.

MATERIALS & METHODS

Our project employs a ML framework to classify SQL queries within web logs as either benign or malicious, designed to enhance web attack detection. The key steps involved are as follows:

Data Collection:

The first step is gathering relevant web server log data, including SQL query entries. These logs are sourced from various web applications and databases, including Kaggle datasets, and are labeled (benign vs. malicious) for training purposes.

Machine Learning Training:

The ML will be trained using the datasets found during data collection. Datasets will have additional columns added indicating true or false for substrings commonly found in SQLi to help the ML make its determinations.

Web Log Classification:

The core of the system is a machine learning model that processes web logs and classifies SQL queries as either benign or malicious. Logs classified as malicious will be flagged for further analysis by security professionals.

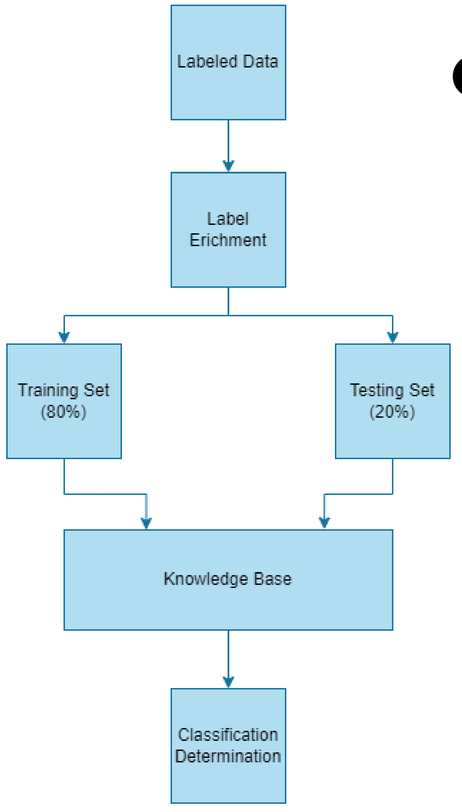

Figure 1. *ML training process*

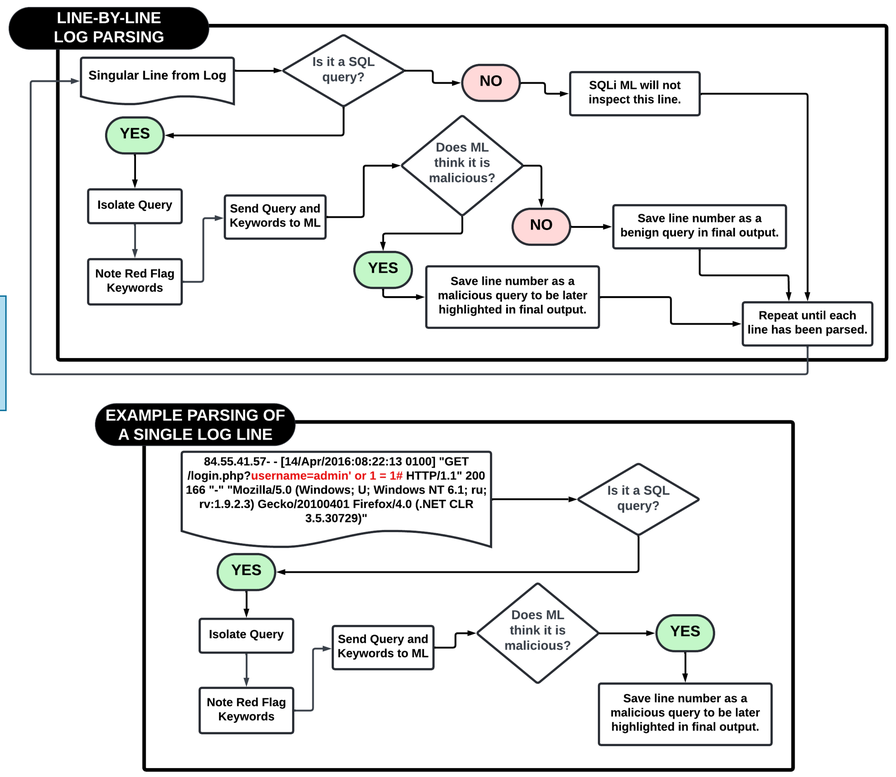

Figure 2. *Log parsing process*

EXPECTED RESULTS

High Accuracy in Classification:

By using a machine learning model to classify the SQL queries, the system will minimize false positives and false negatives, leading to better detection of potential web threats.

User-Friendly Output:

The classified results will be displayed in a straightforward format, making it easy for security professionals to see which logs were determined to be malicious without the need for complex interpretation.

Improved Analyst Efficiency:

By providing classification results and highlighting lines of logs flagged as malicious, the system will reduce the manual effort required by security professionals to interpret raw log data, improving their ability to respond to threats quickly and effectively.

Scalability for Future Web Attacks:

The system should be flexible enough to expand its classification capabilities to recognize additional OWASP Top 10 vulnerabilities and other emerging web attacks with the addition of new ML modules.

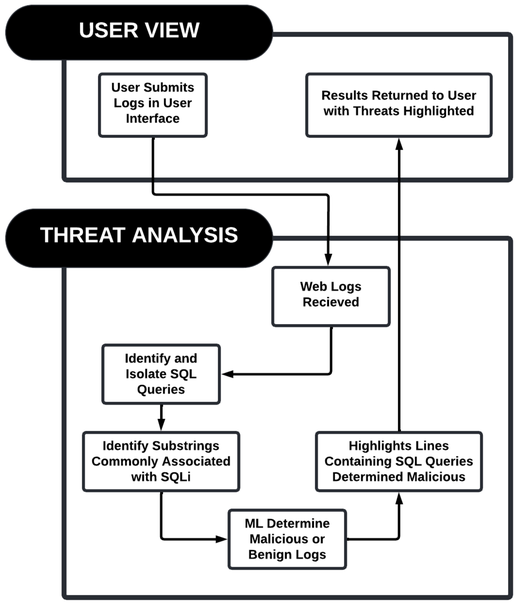

Figure 3. *Expected user experience of ML application*

REFERENCES

Alghawazi M, Alghazzawi D, Alarifi S. "Detection of SQL Injection Attack Using Machine Learning Techniques: A Systematic Literature Review." *Journal of Cybersecurity and Privacy*. 2022; 2(4):764-777. https://doi.org/10.3390/jcp2040039

Muhammad T, Ghafory H. "SQL Injection Attack Detection Using Machine Learning Algorithm." *Mesopotamian journal of Cybersecurity.* 2022; 5-17. https://www.iasj.net/iasj/download/43d7b4e766f39079

Prodromou A. "Using Logs to Investigate – SQL Injection Attack Example." *Acunetix Blog*. 2019; https://www.acunetix.com/blog/articles/using-logs-to-investigate-a-web-application-attack/

ACKNOWLEDGEMENTS

Kaggle user Syed Saqlain Hussain Shah's SQL Injection Dataset was used to train our model and can be found at https://www.kaggle.com/datasets/syedsaqlainhussain/sql-injection-dataset/data