Advanced International Journal for Research
E-ISSN: XXXX-XXXX editor@aijfr.com
Volume 2, Issue 2, March - April 2024

Identifying SQL Injection Vulnerabilities Using SMO Algorithm

B Suganya[1], E C Dhasna[2], K S Sagli[3]

[1,2,3]Research Scholar, Information Technology, K.S.R College of Engineering, Tiruchengode, Tamilnadu, India

Abstract

The proliferation of online applications and services has sparked concerns regarding cybersecurity threats. Among these, SQL injection stands out as a prevalent attack vector exploiting vulnerabilities in online applications to gain unauthorized access to databases. Safeguarding the integrity and security of online systems hinges on the ability to detect and prevent SQL injection attacks. This research employs the Sequential Minimal Optimization (SMO) algorithm to introduce a novel approach for identifying SQL injection attacks in network traffic data. The study proposes a unique methodology that utilizes machine learning to address the critical need for efficient and effective detection methods. Specifically, the research focuses on leveraging the SMO technique to detect and mitigate SQL injection threats using network traffic data. Analyzing the sequence of interactions between hosts, known as network flow data, provides valuable insights into detecting anomalous patterns indicative of potential attack activities.

Keywords: Arduino Uno, Insulated Gate Bipolar Transistor (IGBT), Transformers, Alternating Current (AC), Direct Current (DC), Proteus simulation

1. Introduction

1.1 SQL Injection

SQL Injection poses a significant cybersecurity threat by exploiting vulnerabilities in the database layer of websites or applications. In this type of attack, malicious SQL code is inserted into user inputs such as search bars or login forms. If these inputs are not adequately validated and sanitized, attackers can gain unauthorized access to the underlying database. Once inside, they can manipulate the system, compromise data integrity, or even extract sensitive information. SQL Injection underscores the importance of robust security practices and meticulous coding techniques to prevent breaches and protect against this prevalent form of cyber assault.

1.2 Database Vulnerabilities

Database vulnerabilities represent weaknesses in the digital fortresses that store and manage information. These vulnerabilities act as open doors in cybersecurity, inviting malicious actors to exploit outdated software, weak access controls, or inadequate encryption methods. Essentially, a database vulnerability provides unauthorized parties with opportunities to access, modify, or delete confidential information. To safeguard digital assets, ensure data integrity, and enhance overall system security, it is crucial to identify and remediate

these vulnerabilities. Maintaining resilient database configurations and staying ahead of potential threats necessitate proactive strategies that include regular security assessments and updates.

1.3 SQL Commands

SQL commands serve as the foundational language for relational database operations, empowering users to retrieve, manipulate, and manage data effectively. Structured Query Language (SQL) provides a robust and standardized set of commands that facilitate seamless interaction with database systems. These commands encompass a wide range of functionalities, from simple data retrieval queries to complex operations like data manipulation and schema modification. The versatility of SQL empowers analysts, administrators, and developers to work efficiently with databases, enhancing productivity and enabling sophisticated data handling tasks.

2. Literature Review

2.1 Overview of SQL Injection

Timothy J. has suggested that the rapid expansion of online applications across various domains has paralleled the growing prominence of SQL injection as a cybersecurity concern. As systems become more interconnected, the vulnerabilities associated with inadequate input validation have become increasingly apparent. The ramifications of SQL injection attacks extend beyond compromising individual user accounts as our reliance on web services deepens. These attacks can undermine the fundamental pillars of data confidentiality and integrity in organizational databases. At its core, SQL injection exploits the misplaced trust in user input within online applications, where malicious attackers exploit the failure to thoroughly validate or sanitize user-submitted data to manipulate underlying SQL queries.

2.2 The Evolution of SQL Injection Techniques through History

Oliver Y has proposed that SQL injection techniques have evolved in response to advancing security protections. Attackers have developed more sophisticated methods to inject malicious code into database operations, leveraging features such as stored procedures.

2.3 SQL Injection's Effect on Data Security

Parul Sharma et al. have suggested that successful SQL injection attacks can have far-reaching consequences for businesses beyond the initial breach of security. Financially, data breaches stemming from SQL injection attacks can incur significant costs, including expenses for forensic investigations, legal proceedings, and the implementation of additional security measures. Moreover, the exposure of confidential information can lead to identity theft, financial fraud, and a loss of customer trust, all of which may have enduring negative financial impacts.

2.4 Best Practices and Preventive Measures

Dr. Pooja Raundale has recommended that in addition to examining the impact of SQL injection, significant attention is devoted in the literature to proactive measures that bolster digital defenses. Input validation is a critical defense mechanism that serves as the frontline protection by meticulously scrutinizing user inputs for malicious content. By establishing robust input validation rules, organizations can significantly reduce the attack surface and thwart attempts to inject malicious SQL code. The use of prepared statements and parameterized queries are essential strategies in the ongoing battle against SQL injection.

2.5 Difficulties in Reducing SQL Injection

Muntasir Mamun has highlighted inherent challenges and barriers that make mitigating SQL injection an ongoing challenge. One major obstacle is the presence of legacy code within organizations. Older systems and applications, lacking robust security measures found in modern frameworks, are more vulnerable to SQL injection attacks. Addressing these legacy systems requires considerable resources and expertise, posing practical challenges for businesses seeking to enhance their security posture. Another barrier to preventing SQL injection is a lack of awareness among developers and IT professionals. Despite the availability of preventive measures, critical security practices may be overlooked due to insufficient understanding of evolving attacker tactics. Closing this knowledge gap is essential to ensuring that development teams remain vigilant and proactive in implementing secure coding practices.

2.6 Role of Machine Learning in SQL Injection Detection

Aleksei Shcherbak has put forth researchers' exploration into the use of machine learning (ML) techniques for early detection of SQL injection attempts, given the increasing complexity of cyberattacks. Machine learning algorithms analyze query behaviors and user input patterns to differentiate between legitimate and fraudulent activities. ML holds promise as a proactive defense against evolving SQL injection techniques, leveraging large datasets and trained algorithms to detect anomalous patterns.

2.7 Regulatory Frameworks and Compliance

Rachneet Kaur et al. have proposed that regulatory bodies are taking proactive steps to establish frameworks and compliance requirements in response to the severe repercussions of data breaches caused by SQL injection and other cyber threats. Regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR) compel organizations to prioritize and implement robust defenses. The impact of GDPR, aimed at safeguarding the personal data of European Union citizens, has been extensively studied in the literature.

2.8 Social Engineering Aspects of SQL Injection

S. Saravanan et al. have suggested that the literature examines how social engineering contributes to these exploits, highlighting the complex nature of SQL injection attacks. Attackers often exploit human psychology through psychological manipulation to deceive administrators or users, exploiting human vulnerabilities alongside technological weaknesses. Understanding these tactics is crucial for developing comprehensive security strategies that address both the human and technical aspects of cybersecurity.

2.9 Global Patterns and Trends in SQL Injection Attacks

Lerina Aversano et al. have suggested that the literature offers a macroscopic view that illuminates the evolving landscape of cyber risks by meticulously analyzing global patterns and trends in SQL injection attacks. One significant observation is the geographic dispersion, demonstrating varying frequencies of SQL injection attacks across different regions. Researchers often attribute these differences to infrastructure vulnerabilities, disparities in cybersecurity awareness, and the overall digital maturity of nations.

2.10 Educational Initiatives and Awareness Programs

SURA MAHMOOD ABDULLAH et al. have proposed that certain literature emphasizes the effectiveness of educational initiatives and awareness campaigns in mitigating SQL injection attacks, recognizing the pivotal

role of education in fortifying digital defenses. These programs typically target developers, who play a crucial role in the development and maintenance of software systems.

2. Existing System

SQL injection attacks pose a significant security threat to web applications by granting attackers unrestricted access to databases containing potentially sensitive information. Despite numerous solutions proposed by researchers and practitioners to address SQL injection, current options either fail to completely resolve the issue or suffer from limitations that hinder their adoption and effectiveness.

3. Proposed System

Our proposed method represents an advanced defense strategy against the escalating threat posed by SQL injection attacks in today's dynamic web application landscape. Employing the Sequential Minimal Optimization (SMO) technique, our approach is designed to identify and mitigate SQL injection risks within network traffic data. Recognizing the critical need for robust and efficient detection mechanisms, our method harnesses the inherent capabilities of machine learning. By analyzing network flow data, which logs interactions between hosts, our approach identifies suspicious patterns indicative of SQL injection attacks. A notable feature of the SMO algorithm is its proficiency in handling extensive and intricate datasets, facilitating swift detection with enhanced accuracy and minimized false positives.

4. Modules

4.1 Data Collection

The data collection phase comprises three modules: the query tree log collector, regular query generator, and malicious query generator. The query tree log collector module gathers query trees generated by the PostgreSQL database system. A set of standard SQL queries is created by the regular query generator module for training the SVM classification algorithm. To evaluate the framework's ability to detect SQL Injection Attacks (SQLIAs), a collection of malicious SQL queries is generated by the malicious query generator module. Subsequently, the proposed method employs a unique approach outlined in the study to transform query trees into an n-dimensional feature vector. This process involves retrieving syntactic and semantic characteristics and using various statistical models to convert these features into numerical values.

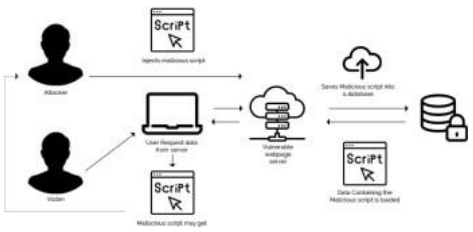

FIGURE 5.1.

5.2 Data Preprocessing

The data preprocessing module of the proposed framework consists of three components: vector generator, feature extractor, and feature transformer. The feature extractor component extracts syntactic and semantic characteristics from the query tree logs collected by the query tree log collector module. This includes data types, table connections, and query structure, ensuring relevant information is gathered and utilized in

subsequent stages to distinguish between malicious and legitimate SQL queries. The extracted characteristics are then converted into numerical values by the feature transformer component, which are suitable for input into the SVM classification algorithm, as SVMs require numerical inputs.

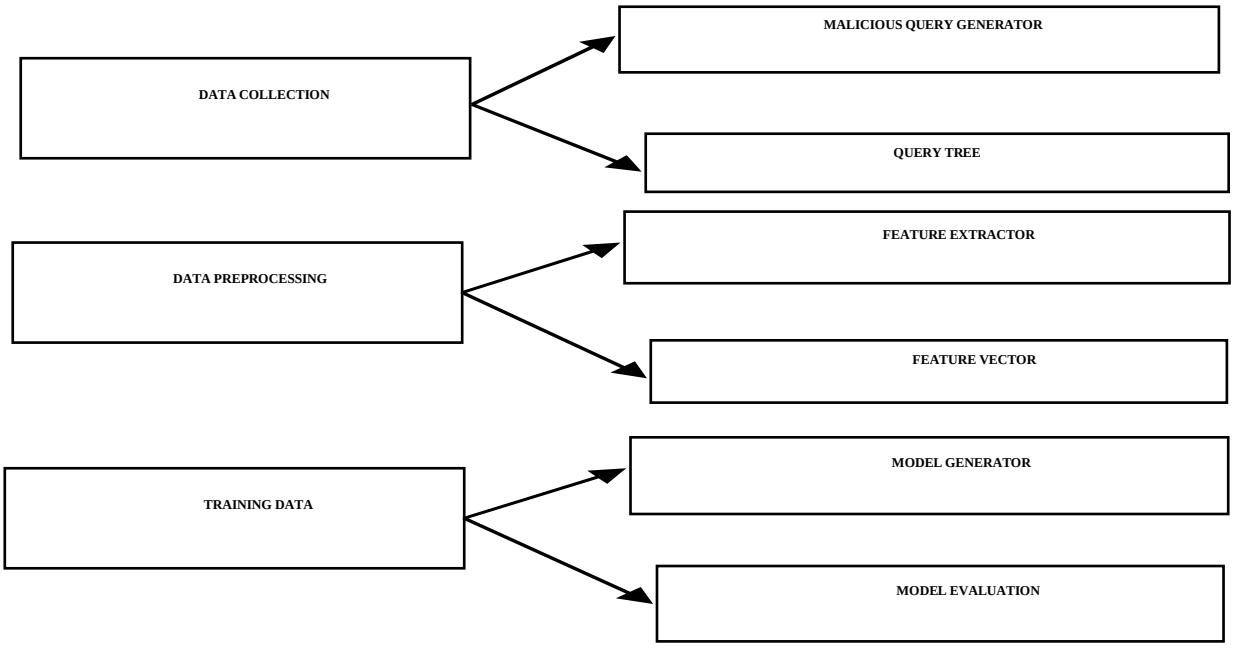

Figure 1. block diagram

5.3 Training Data

The feature vectors generated in the data preprocessing module are utilized by the model generator component to train the SVM classification algorithm. Specifically, the model generator trains the SVM algorithm to identify normal queries using feature vectors from the regular query generator module. Subsequently, the trained SVM model is applied to classify incoming SQL queries as either malicious or legitimate. The performance of the trained SVM model is evaluated by the model evaluator component, specifically assessing its ability to detect SQLIAs using feature vectors generated by the malicious query generator module.

5.4 Attack Detection

The SQLIA classifier is responsible for classifying incoming SQL queries as malicious or legitimate during the attack detection phase. It utilizes the feature vectors generated by the vector generator component in the data preprocessing module, along with the SVM model trained during the training phase. Initially, the input SQL query is converted into a query tree structure by the feature extractor, feature transformer, and vector generator components. This feature vector is then fed into the SVM model to determine if the query exhibits malicious behavior. The detection phase of the proposed framework relies on the SQLIA classifier as its primary component for categorizing SQL queries effectively, aiming to minimize false positives by leveraging the trained SVM model and feature vectors generated during data preprocessing.

6. Result and Discussion

the experimental evaluation of the proposed sql injection attack detection framework using real-world network traffic data and the sequential minimal optimization (smo) method yielded promising results. the system demonstrated high efficacy in accurately detecting and mitigating sql injection attacks, as evidenced by its elevated recall rate and precision. leveraging the smo algorithm's ability to handle complex, multidimensional data significantly enhanced detection accuracy and minimized false positives.

algorithm	accuracy
Existing system	75
Proposed system	81

Table 1. Comparison table

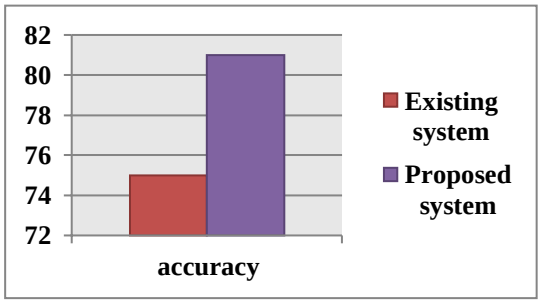

Figure 2. Comparison grap

table 1 and figure 2 present a comparative analysis of the accuracy between the existing system and the proposed system. the results indicate a notable improvement in accuracy with the proposed method achieving 81%, compared to 75% with the existing system. this substantial increase underscores the effectiveness of the proposed algorithm in addressing the challenges or limitations encountered by the current system. by incorporating innovative features, algorithms, or techniques, the proposed system demonstrates potential for providing a more reliable and efficient solution in detecting sql injection attacks. these findings mark significant progress in the field and suggest possibilities for enhancing system robustness and performance through advanced algorithmic enhancements.

7. Conclusion

In summary, the developed system for detecting SQL injection attacks (SQLIAs) represents a significant advancement in safeguarding database-driven websites from malicious intrusions. The system seamlessly integrates SVM classification, multidimensional sequences, and an advanced feature extraction methodology, demonstrating remarkable accuracy in identifying SQL injection attacks at the database level. Rigorous testing using PostgreSQL's internal query trees validates the robustness of the approach, achieving a detection rate of over 99.6% with minimal false positives. Besides addressing limitations of current application-level detection techniques, the proposed methodology offers a practical and efficient solution for real-world deployment. Its effectiveness in fortifying the security posture of database systems positions it as a valuable tool in the ongoing battle against evolving cyber threats targeting critical data repositories.

8. Future Work

Future research endeavors could explore enhancing the flexibility of the SQL injection attack (SQLIA) detection framework to accommodate a broader array of database management systems and platforms. Additionally, refining the feature extraction process to adapt to evolving syntactic and semantic attributes of SQL queries will further bolster the system's resilience against emerging attack vectors.

References

1. MartinsN, CruzJ.M, CruzT, AbreuP.H. "Application of Adversarial Machine Learning in Intrusion and Malware Scenarios: A Comprehensive Review," IEEE Access, 2020, 8, 35403–35419.
2. Tang P, Qiu W, Huang Z, Lian H, Liu G. "Artificial Neural Network-Based Detection of SQL Injection," Knowledge-Based Systems, 2020, 190, 105-528.
3. Marashdeh Z, Suwais K, Alia M. "Survey on SQL Injection Attacks: Detection Challenges," Proceedings of the International Conference on Information Technology (ICIT), Amman, Jordan, 2021, 957–962.
4. Mejia-Cabrera H.I, Paico-ChilenoD, Valdera-Contreras J.H., Tuesta-Monteza V.A., Forero M.G."Machine Learning for Automatic Detection of Injection Attacks in NoSQL Databases," Springer, Berlin/Heidelberg, Germany, 2021, pp. 23–32.
5. Sivasangari A. "Machine Learning Algorithm for SQL Injection Attack Detection," Proceedings of the 5th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, June 3–5, 2021, pp. 1166–1169.
6. Siddiq M.L., Jahin R.R., Rafid M., Islam U. "SQLIFIX: Learning-Based Approach to Fix SQL Injection Vulnerabilities in Source Code," IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, March 9–12, 2021, pp. 354–364.
7. Crespo-Martínez I.S., Campazas-Vega A., Guerrero-Higueras A.M., Riego-DelCastillo V., Alvarez-Aparicio C., Fernández-Llamas C. "Detection of SQL Injection Attacks in Network Flow Data," Computers & Security, vol. 127, p. 103093, 2023.
8. Wang Y.-C., Zhang G.-L., Zhang Y.-L."Analysis of SQL Injection Based on Petri Net in Wireless Network," Journal of Information Science & Engineering, vol. 39, no. 1, 2023.
9. Kumar M. "SQL Injection Attack on Database Systems," Wireless Communication Security, p. 183, 2023.
10. Baklizi M., Atoum I., Hasan M.A.-S., Abdullah N., Al-Wesabi O.A., Otoom A.A. "Prevention of Website SQL Injection Using a New Query Comparison and Encryption Algorithm," International Journal of Intelligent Systems and Applications in Engineering, vol. 11, no. 1, pp. 228-238, 2023.
11. Yadav N., Shekokar N.M."Case Study on SQL Injection Attacks on Indian Websites," in Cyber Security Threats and Challenges Facing Human Life: Chapman and Hall/CRC, 2023, pp. 153-170.
12. Hadabi A., Elsamani E., Abdallah A., Elhabob R."Efficient Model for Detection and Prevention of SQL Injection Attacks," Journal of Karary University for Engineering and Science, 2022.
13. Manhas S. "Comprehensive Analysis of SQL Injection Attacks," in Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2022, Volume 1: Springer, 2022, pp. 3-12.
14. Dhanushya K., Faritha Banu M., Tamilzharasu M.P., Suganya S. "Blood Donor App," Vol. 14 No. 1 (2023).
15. Bhuvanesh G., Gopinath N., Sharvesh M., Suganya S. "Detection and Classification of Rice Leaf Diseases Using Image Processing," Vol. 14 No. 1 (2023).