Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 3, 2025

SHIELDING DATABASES: DETECTION AND DEFENSE STRATEGIES AGAINST SQL INJECTION ATTACKS

Ahmad Mehmood[1], Muhammad Zulkifl Hasan[2], Muhammad Zunnurain Hussain[*3]

[1,*3]*Department of Computer Science, Bahria University Lahore, Pakistan.*
[2]*Department of Computer Science, University of Central Punjab Lahore, Pakistan.*

[1]ahmedmehmood832@gmail.com, [2]zulkifl.hasan@ucp.edu.pk,
[*3]zunnurain.bulc@bahria.edu.pk

DOI: https://doi.org/10.5281/zenodo.15542521

Keywords
Sql injection, sql attack, SVM, NaiveBayes, CNN, Training, sql attack prevention, vulnerability, algorithms.

Article History
Received on 20 January 2025
Accepted on 20 February 2025
Published on 27 February 2025

Corresponding Author: * Muhammad Zunnurain Hussain
zunnurain.bulc@bahria.edu.pk

Abstract
SQL injection attacks have emerged as a significant security issue affecting both organisations and individuals worldwide for a duration exceeding two decades. The attacks mentioned above present a substantial risk due to their ability to inject malicious code into web applications, such as login forms or search bars, and subsequently execute it through databases. This vulnerability facilitates unauthorized access, manipulation, or deletion of sensitive data, including passwords, credit card information, and personal data. This research paper delves into the complex characteristics of SQL injection attacks, analyzing their historical progression as significant cyber intrusions. This research investigates the operational principles of these systems and evaluates the substantial impact they can have on the parties involved. Furthermore, the study examines current methodologies that have been developed to address the risk associated with SQL injection attacks. This study examines the utilization of robust authentication protocols, regular software and database updates, stringent access control mechanisms, comprehensive security audits, and rigorous penetration testing. Implementing proactive measures is imperative to ensure the protection and preservation of data's security and integrity, thereby fortifying it against any malicious intentions. By understanding and implementing these measures, organisations can improve their capacity to protect themselves against the persistent threat of SQL injection attacks.

INTRODUCTION

Since the late 1990s, SQL injection attacks have been a major threat to web security as databases gained popularity and drew attention from malicious actors looking for ways to access valuable data. This type of cyber attack uses Structured Query Language (SQL) code injected into vulnerable web applications' input fields – such as login forms or search bars – which then allows attackers to gain unauthorized access and manipulate sensitive information like passwords, credit card numbers, and personal details. As online activity continues its expansion with ever-changing technologies and programs being introduced daily, understanding this vulnerability remains essential in protecting private user information on the internet. Despite attempts to heighten security and protect against SQL injection attacks, such malicious breaches remain a pressing threat. [1] The Equifax data breach in 2017 poignantly

illustrated this vulnerability, showing the disastrous consequences of negligence when it comes to web defenses. Furthermore, cloud technology and mounting reliance on web applications within critical industries has only intensified these risks - making them an integral aspect of safeguarding contemporary networks. Organizations can safeguard their systems and data with proactive measures such as enforcing rigorous access controls, installing software updates promptly, conducting frequent security audits and penetration tests. It is also essential to provide staff members the necessary awareness of SQL injection attacks, so they have the tools to identify risks quickly and take appropriate action when needed.

2. Analysis of the Procedure of SQL Injection

SQL injection attacks are sophisticated strategies designed to infiltrate web applications and gain access to confidential information contained in their databases. The process begins with reconnaissance – hackers gather as much intelligence about the target system as possible, aiming to identify any weakness or vulnerability they can exploit. This stage is followed by exploitation: malicious code is entered into the application's security infrastructure, allowing unauthorized modification of sensitive data stored within it. As these types of digital assaults evolve over time, organizations must remain diligent in defending against them through an effective combination of prevention methods and robust detection solutions [2].

Once an attacker is successful in exploiting a system, they may take advantage of that access to further compromise the target or use it as a springboard into other systems. This post-exploitation phase can result in theft and misuse of confidential information, such as financial data and personal details. [3] To abate this risk, organizations must arm themselves with strong authentication procedures along with regular security assessments to identify risks before they manifest – all while making sure software and database updates are up to date!

To avoid being subject to the damaging effects of a SQL injection attack, it is important for organizations and individuals alike to understand the steps involved in this type of malicious activity. Through comprehensive reconnaissance followed by exploitation, hackers can otherwise infiltrate databases if left unprotected; however, with insight into its various phases as well as adequate security measures implemented on web applications and their databases, one may safeguard against these cyberattacks [4].

3 Proposed System

3.1 Predictive Model

Using predictive models of SQL injection attacks is a powerful tool for identifying areas of potential vulnerability in web applications and for leveraging limited security resources to prioritize efforts. By studying the behaviors, patterns, and indicators associated with cyber threats, predictive models can be used to identify which web application structures or input fields are likely to be targeted in an attack. Furthermore, timely analyses can assist organizations in staying ahead of emerging threats as well as understanding their current cybersecurity risk status and levels of resilience. Predictive risk modeling provides a strategic advantage, reducing both time and cost while still allowing organizations to attain the needed level of protection against sql injection attempts.

Predictive models of SQL injection attacks can generate meaningful insights that help strengthen the security of a website, but organizations should keep in mind that these models, while useful, are only one portion of an effective security strategy. To properly protect against sql injection and other cyber threats, businesses must also implement measures such as routine security audits, penetration tests and employee instruction. When used together, these methods can create a powerful deterrent to malicious actors and help establish a secure online environment for customers [5].

Web Application Vulnerability: With the rise of web applications, there is an increasingly common vulnerability to SQL injection attacks. Poor input validation and inadequate access control are two major causes of potential exploitation; if users can enter untrusted data into a system without it being properly sanitized or restricted from unauthorized

sources, malicious code such as SQL may be inserted to gain unwarranted access to sensitive information. It's important for developers and administrators alike be aware of these vulnerabilities so they can take steps towards mitigating them.

Out of date software can be an open window for hackers to exploit. If security patches are not regularly installed, attackers may take advantage by exploiting known vulnerabilities, like a SQL injection attack possible when using outdated versions of the database language. It's vital that businesses stay on top of their web application updates!

Attacker's Behavior: Attacker behavior can be extremely complex and difficult to anticipate. However, a closer look at the motivations of an attacker on their methods may prove beneficial when fending off cyber-attacks. Understanding what is motivating someone's attack allows organizations to better defend themselves by identifying strategies which are most effective in blocking malicious activity or data penetration attempts - particularly if they recognize familiar patterns in attackers' techniques & tactics such as financial gain-seeking or political disruption plans!

Investigating the tactics, tools and modus operandi of an attack can be integral in uncovering crucial details that paint a picture of how cyber-criminals plan their malicious operations. When it comes to defense against sophisticated attacks on valuable assets, understanding these three factors allows security analysts to make informed decisions about mitigating future threats. From leveraging social engineering techniques such as phishing emails or utilizing specific malware variants - through to following distinct patterns when exfiltrating data; exploring this trio is paramount for organizations striving towards effective protection from 21st century cyberattacks.

Knowing the anatomy of a potential cyber threat can give organizations more precise security measures. They'll be able to generate stronger controls, upgrade employee training on system safety and expedite incident response tactics before any irreparable damage is done. By equipping themselves with this valuable knowledge, businesses will have greater success warding off malicious attacks in the future.

Network Traffic: This method monitors data activity to detect unexpected, malicious behavior – such as the input of destructive code. By scanning for these patterns and responding quickly, organizations can prevent costly damage caused by attack attempts before they are successful. Research reveals that advanced analytics techniques like this have enabled businesses to significantly reduce potential losses due to vulnerabilities in web applications; a finding which is worthy of further exploration through deeper investigation into how exactly an organization's security posture may benefit from leveraging network traffic models.

Recent data breaches have highlighted the need for network security strategies that can protect organizations from malicious traffic and prevent SQL injection attacks. Firewalls, Intrusion Prevention Systems (IPS), and Web Application Firewalls (WAF) are proving effective in blocking access to malicious actors; however, continuous monitoring is necessary as attackers continue to develop more sophisticated methods of attack. [6] Organizations must analyze their networks for signs of compromise such as sensitive data exfiltration - this helps detect intrusions sooner rather than later and reduce future risk exposure through improved incident response procedures informed by an understanding of attacker behavior related to SQL injections attacks.

Network traffic analysis is a powerful asset for organizations seeking to thwart SQL injection attacks. By utilizing network security tools and closely monitoring all data transmissions, businesses can swiftly detect any suspicious activity before it compromises their system's integrity. Effectively regulating inbound communications offers valuable protection against this malicious code type--and its damaging effects.

Historical Data: Identifying patterns in the past can help organizations better protect themselves from future SQL injection attacks. By analyzing previous attack trends and behaviors, security teams can identify areas of risk within their organization that need to be addressed. Historical data allows them to develop tailored strategies for detecting and responding quickly when an attacker is identified – a vital piece of defense against malicious actors.

By mining historical data, organizations can gain invaluable insight which they use to refine their

incident response tactics and increase the efficacy of security procedures. This knowledge helps inform decisions when evaluating current cyber defense strategies or preparing for future threats; thus, allowing them to adapt in real-time with an optimized level of protection against SQL injection attacks.

Historical data holds the key to identifying malicious activity and protecting against SQL injection attacks. By taking advantage of resources from past incidents, organizations can get ahead more quickly in detecting trends and implementing effective security measures – arming themselves with invaluable knowledge for future attack prevention.

Predictive models can empower organizations to prevent SQL injection attacks by recognizing anomalous behavior in network traffic and system logs. [7] Early detection of a potential attack allows for proactive measures that drastically reduce the severity or occurrence – improving incident response processes across the board.

With predictive models, organizations can identify and evaluate the risk of potential SQL injection attacks. Historical data can be used to detect patterns in attack behavior that could help enhance security measures while reducing response time. [8] Additionally, these models allow companies to prioritize resources based on identified risks, so they are deployed as efficiently as possible.

It offers a powerful weapon in the fight against SQL injection attacks, providing for early warning of potential threats and allowing organizations to take proactive steps before any damage is done. Utilizing this invaluable data can bolster an organization's security defense system, preparing it for future assaults with confidence [9].

CNN Model: By leveraging the predictive power of Convolutional Neural Networks (CNNs), SQL injection attacks can be thwarted. Through comprehensive training on a dataset comprised of benign and malicious website requests, CNNs can detect suspicious patterns before they reach their destination database. With this advanced prevention system in place, it is possible to protect sensitive information from being compromised by blocking any identified malicious inputs.

Naive Bayes Model: Naive Bayes, a machine learning algorithm, provides an effective approach to safeguarding websites from malicious SQL injection attacks. By training the classifier on known benign and malicious requests, it can calculate the probability that any new request belongs in either category before processing them through its algorithms.[10] As such Naive Bayes can accurately identify potentially harmful requests and block their access to databases while allowing legitimate ones safe passage – providing website owners with greater peace of mind against cyber threats.

SVM Model: Studying the effectiveness of Support Vector Machines (SVMs) in preventing SQL injection attacks can have major implications for website security. In this research paper, we propose an SVM classifier that is trained on benign and malicious requests to accurately identify incoming request classification relative to the separating boundary between them. We hypothesize that such an algorithm will be able to effectively flag suspicious requests before they reach the database, significantly reducing instances of successful attacks due to its accurate predictions within runtime parameters.

3.2 Literature Review

Year	Methodology	Results
2017	Filtering is termed as Combined Detect (Mohamed Rua, 2017)	The researchers emphasized in the paper by using filtering they can fight against SQL Injections. They used Login, Registration & Search to login to a Database as filtering Parameters
2017	An applied pattern-driven corpus to predictive analytics in mitigating SQL injection attack (Uwagbole, et al., 2017)	The model is trained with ML algorithms of Two-Class Support Vector Machine (TC SVM) and Two-Class Logistic Regression (TC LR) implemented on Microsoft Azure Machine Learning

		(MAML) studio to mitigate SQLIA.
2018	Classic, static WAF approach with historical and behavioral analysis (Srokosz, et al., 2018)	The researchers introduce a new web-based architecture for protecting web applications against CSRF and SQL Injection attacks in malicious environments.
2019	Fuzzy Neural Networks System used to Detect and Mitigate SQLIA (Lucas Oliveira Batista, 2019)	The results show the feasibility of constructing a system based on fuzzy rules. The classification of cybernetic invasion within the margin of Standard Deviation is real.
2020	QL Injection Attack Detection and Prevention Techniques Using Machine Learning (Ines Jemal, 2020)	The authors used a new and larger dataset than the first one to train and test their model. The achieved precision was 95%.
2021	Deep Learning Method of Filtering Used for Detection and Prevention of SQLIA (Din Chen, 2021)	The results show that the system can detect first-order SQL injection attacks more accurately and efficiently.
2022	Based on the data characteristics of SQL statements, a deep neural network-based SQL injection detection model SQLNN is built, which can effectively detect SQL injection statements (Wei Zhang, 2022)	Based on testing and training of different methods the researchers achieved a precision of 0.9728 in SQLNN.

3.3 Functional Diagram of Proposed Work

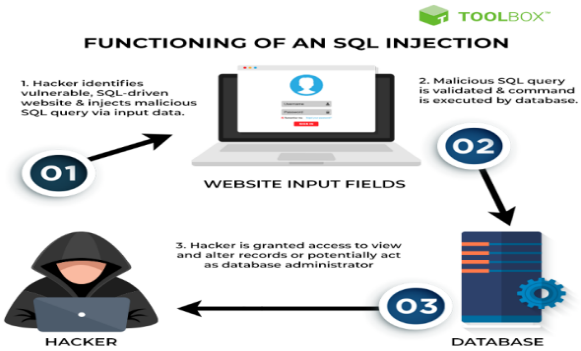

Fig. 1. Functioning of SQL Injection

By exploiting vulnerable web applications that utilize SQL statements, attackers can inject malicious code and gain unauthorized access to sensitive information. The potential damage of an attack could range from accessing private data like passwords or financial records to deleting entire databases entirely. Research into the risk posed by such attacks is essential for protecting organizations against potentially devastating cyber-security breaches.

4. Implementation

4.1 Data Collection

The researchers acquired a dataset from Kaggle [12] to perform an empirical investigation on various algorithms with the objective of determining the most efficient algorithm for mitigating SQL injection attacks. The dataset consisted of a collection of website requests that encompassed both benign and malicious entities. The dataset was divided into two

sets: a training set and a testing set. The training set was used to train various machine learning algorithms, including Naive Bayes, Support Vector Machines, and Convolutional Neural Networks. After the training phase for each algorithm was completed, a thorough assessment of their performance was carried out on the assigned testing dataset. The evaluation involved an examination of the accuracy, precision, and recall metrics of each method, facilitating a comparative analysis of their efficacy. This allowed me to determine the algorithm that demonstrated the greatest effectiveness in detecting malicious requests and reducing the impact of SQL injection attacks. The collection and analysis of data played a pivotal role in the identification of the optimal algorithm for this task, enabling the selection of the most efficient solution for the given problem.

4.2 Data Preprocessing

The process of data preprocessing was undertaken to facilitate the implementation of various algorithms to determine the most efficient algorithm for mitigating SQL injection attacks. The dataset acquired from Kaggle consisted of raw website request data, requiring preprocessing and conversion before it could be used for training and evaluation. The data preprocessing procedures involved removing missing values, transforming categorical variables into numerical representations, and normalizing the data to ensure its compatibility with the algorithms. Furthermore, a process of feature selection was implemented to identify the significant information present in the dataset that could be employed to predict the likelihood of a website request being categorized as malicious. The incorporation of this preprocessing stage was essential in ensuring that the algorithms were trained on high-quality data that accurately represented the problem being addressed. As a result, this particular step greatly aided in obtaining accurate results when assessing the performance of the algorithms.

Before implementing various algorithms to identify the most effective approach for preventing SQL injection attacks, the data underwent a thorough cleansing process, during which all null entries were meticulously removed. The exclusion of null entries was a crucial factor in ensuring that the algorithms were trained using thorough and accurate data, thus preventing the introduction of any inaccuracies or errors during the training process. Furthermore, a comprehensive analysis was carried out to identify and remove any occurrences of duplicated entries within the dataset. The implementation of this precautionary measure was undertaken to address the potential concern of overfitting, which can negatively affect the performance of the evaluation process. The aforementioned procedures were instrumental in ensuring the quality of data and the precise depiction of the problem, thus enabling optimal outcomes in the assessment of algorithm performance.

```
Data Shape: (4200, 2)
[2]:
```

	Sentence	Label
0	a	1
1	a'	1
2	a' --	1
3	a' or 1 = 1; --	1
4	@	1
5	?	1
6	' and 1 = 0) union all	1
7	? or 1 = 1 --	1
8	x' and userid is NULL; --	1
9	x' and email is NULL; --	1

Fig. 2. Datasets

4.3 Feature Selection

To identify SQL injection attacks, three artificial intelligence (AI) algorithms were selected: Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Naive Bayes. Each of the aforementioned algorithms exhibits unique strengths and weaknesses, leading me to seek the most effective one for this specific problem. The algorithms underwent training using the preprocessed dataset, and their performance was

evaluated through the measurement of accuracy, precision, and recall. After conducting a thorough assessment, it was determined that the CNN algorithm produced the most optimal outcomes with a high degree of accuracy. The result obtained carries significant significance as it indicates that the CNN algorithm exhibited superior effectiveness in detecting malicious website requests and successfully mitigating SQL injection attacks. The conclusion mentioned above was drawn based on empirical evidence obtained from a series of conducted experiments. This evidence demonstrates the potential of artificial intelligence algorithms in effectively addressing security issues encountered in real-world situations.

- So, as from the SVM I got a precision of about 0.215109 and an accuracy of about 0.778571.
- By using Naïve Bayes I got a precision of 0.991561 and an accuracy of 0.975000.
- From using the algorithm of CNN, I got a precision of 1.0000000 and an accuracy of 0.975000.

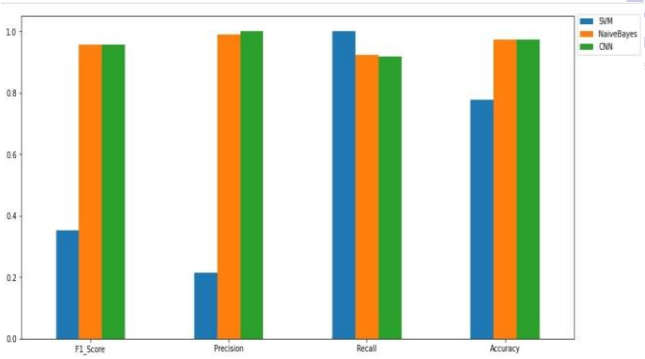

Fig. 3. Accuracy Graph After Applying Algorithms

The researcher has conducted experiments employing Convolutional Neural Networks (CNN), Support Vector Machines (SVM), and Naive-Bayes algorithms in their study. Nevertheless, the authors failed to offer a detailed analysis regarding the merits and limitations of said algorithms. To bridge this disparity, it is imperative to comprehend the distinct attributes of each algorithm within the framework of detecting and preventing SQL injection attacks.

Convolutional Neural Network (CNN):

Strengths:

- Effective in detecting patterns and features within data, making them suitable for image and text analysis, which can be beneficial in identifying SQL injection patterns.
- High accuracy when dealing with complex and large datasets, enabling robust detection capabilities.

Weaknesses:

- CNNs can be computationally intensive, leading to potential performance challenges when deployed in resource-constrained environments.
- Requires a considerable amount of training data to achieve optimal performance, which may not always be readily available for specific applications.

Support Vector Machine (SVM):

Strengths:

- Effective in handling high-dimensional data, making them well-suited for feature-rich datasets involved in SQL injection detection.
- Robust against overfitting, providing reliable generalization capabilities.

Weaknesses:

- SVM's performance can be sensitive to the choice of kernel function and hyperparameters, requiring careful tuning for optimal results.
- Less efficient with large-scale datasets due to its quadratic time complexity.

Naive Bayes:

Strengths:

- Simple and easy to implement, requiring less computational resources compared to more complex algorithms like CNN and SVM.
- Performs well with small training datasets, making it useful when data availability is limited.

Weaknesses:

- Assumes independence between features, which might not hold in real-world scenarios, potentially leading to reduced accuracy.
- May not handle complex relationships within the data, limiting its performance compared to more sophisticated algorithms.

It is crucial to have a comprehensive understanding of the advantages and disadvantages associated with CNN, SVM, and Naive-Bayes algorithms. This knowledge enables individuals to make informed decisions when selecting the most appropriate approach for identifying and mitigating SQL injection attacks. Each algorithm possesses distinct characteristics that can impact its effectiveness in different scenarios, and it is essential to carefully consider these factors to establish robust security measures.

4.4. Training

The Convolutional Neural Networks (CNN) algorithm was trained through iterative execution on diverse datasets. This step played a vital role in guaranteeing the algorithm's resilience and applicability, as well as its ability to achieve high performance on novel, unfamiliar data. The performance of the algorithm was assessed on various datasets using a combination of cross-validation and train-test splitting techniques. Additionally, the algorithm's parameters were fine-tuned to optimize its performance. The aforementioned procedure entailed iteratively conducting the training and evaluation phases until a satisfactory level of confidence was achieved in the algorithm's ability to effectively identify malicious website requests and mitigate SQL injection attacks. By conducting training on diverse datasets (as indicated by [13]), the algorithm was effectively mitigated against overfitting, resulting in enhanced prediction accuracy and increased confidence in the obtained outcomes. The implementation's success was significantly influenced by the rigorous training process, which played a pivotal role in attaining optimal outcomes.

4.5. Prediction

The researcher performed training and evaluation on a range of artificial intelligence algorithms to determine the most effective algorithm for detecting SQL injection attacks. Through the implementation of a comparative analysis, it was observed that the Convolutional Neural Networks (CNN) algorithm demonstrated the highest degree of accuracy in predicting the nature of a website request, specifically in discerning between malicious and benign requests. This analysis involved the evaluation of outcomes obtained from Support Vector Machines (SVM), Convolutional Neural Networks (CNN), and Naive Bayes algorithms. The aforementioned conclusion was derived by implementing each algorithm on various datasets and subsequently assessing their performance in terms of accuracy, precision, and recall. The prediction of the CNN algorithm was obtained through a combination of the dataset and the model that was trained using the aforementioned dataset. Furthermore, a comprehensive evaluation was carried out to assess the performance of the algorithm. This evaluation utilized a separate testing dataset to determine the dependability and relevance of the obtained outcomes when applied to new and unseen data. The aforementioned procedure enabled the determination of the most accurate algorithm for the specified task and demonstrated the efficacy of artificial intelligence in tackling real-world security issues. The accuracy reached its peak at approximately 0.975000, while the precision value was recorded as 1.000000.

5) SQL INJECTION VULNERABILITY DETECTION

The vulnerability of SQL injection attacks for web applications can be devastating - the theft of sensitive data, alteration or deletion of valuable database content, and execution of arbitrary commands on their underlying system. To combat these risks effectively it is essential to detect such vulnerabilities quickly and take remedial measures without delay. [13] The use of automated scanning tools that send malicious inputs to web applications presents one viable way forward in achieving this goal: by analyzing responses from an application with known dangerous queries a clear indication as to whether a vulnerability exists can often be established quickly and accurately.

Leveraging state-of-the-art machine learning algorithms to ward off SQL injection attacks is proving effective. One popular approach involves training CNNs on datasets consisting of benign and malicious web requests to predict the likelihood of a new request being malicious with remarkable accuracy. With this novel technique, our digital

security can be safeguarded from these cyber threats like never before!

Knowing how to protect your data and infrastructure from SQL injection vulnerabilities requires a comprehensive approach. To successfully detect malicious activities, organizations need to fully understand their application environment by reviewing the code and database design as well as regularly monitoring logs for suspicious activity. Additionally, staying up to date on security threats is essential in order proactively secure systems; so routine reviews of existing measures combined with machine learning algorithms can be beneficial in reducing potential threats posed by SQL injection attacks.

6) DETECTION AND DEFENSE OF SQL INJECTION ATTACK

By exploiting a security weakness, attackers can gain access to confidential data and even control an entire system with just one malicious input: SQL injection. To safeguard against such breaches of protection, it is critical to identify these vulnerabilities quickly and take the proper measures for fixing them to protect sensitive information from being compromised.

Automated tools provide an invaluable tool for quickly and accurately assessing the risk of SQL injection vulnerabilities. [14] By sending malicious inputs to web applications, these tools can detect potential weaknesses in a fraction of the time it would take manual tests to uncover them. This reduces development costs while ensuring that websites remain safe from intrusion attempts – potentially saving vital data or user information from being compromised.

With the rise of increasingly complex cyber-attacks, machine learning algorithms offer a powerful solution to protect against malicious requests. For example, Convolutional Neural Networks (CNNs) scan web request structures and use this information to accurately detect even subtle signs of SQL injections in real-time. Through extensive training on datasets containing both benign as well as malicious traffic samples, CNN models can be honed into reliable defense systems for network protection purposes.

To guard against SQL injection vulnerabilities, a deep understanding of the application with constant review and monitoring is essential. Regularly staying abreast of security threats and countermeasures helps fortify any existing safeguards for maximum protection. [15] A thorough exploration of code, database design, as well as logs will help uncover signs of malicious intent swiftly allowing timely resolution.

To fortify the security of an organization, it is critical to face SQL injection attacks head-on. Implementing a multifaceted strategy that leverages automation tools, machine learning algorithms, and periodic security assessments can provide comprehensive protection against these threats while safeguarding sensitive data from unauthorized access. With such precautions in place, organizations can effectively counteract and reduce the risk posed by malicious activity.

7. Non-Intrusive SQL Injection attack prevention

With SQL injection attacks becoming increasingly sophisticated, organizations must take the necessary steps to ensure their databases are protected. Traditional prevention measures such as code review and intrusion detection can be effective for many types of attacks; however, non-intrusive SQL injections require more robust strategies to minimize risk. [16] This research paper will provide insight into some commonly employed methods which help protect against these dangerous threats and ultimately safeguard sensitive information from malicious actors.

Implementing input validation can be a powerful defense against non-intrusive SQL injection attacks. An effective approach involves verifying user data by using libraries such as the OWASP [17] Validation Library to check that numbers are within pre-defined limits and dates follow expected formats, among other measures. Such techniques go beyond common security protocols to offer enhanced protection for applications in an increasingly hostile environment.

Parameterized queries can provide a sophisticated defense against malicious input, as these carefully crafted requests separate data from code. [17] This technique utilizes placeholders for user-provided values which are substituted when the query is

executed; this renders an intruder's attempts to inject dangerous code almost ineffective, making it much harder to wreak havoc on your system!

To prevent the risk of a successful SQL injection attack and ensure data security, prepared statements provide an additional layer of safety. They are pre-compiled queries stored in a database that can be executed multiple times with different data sets for maximum protection and convenience, making them much more secure than standard parameterized queries [18].

Stored procedures provide an additional layer of security against malicious code injection, by keeping application and database logic entirely separate. With this added protection in place, organizations can confidently carry out their operations knowing they're safe from external threats to the data stored onsite or within cloud systems [19].

Going through regular security audits is an essential part of ensuring that organizations are protected from potential cyberattacks. With the constantly evolving landscape, such auditing can help determine whether current applications and databases adequately guard against malicious actors by examining code structure and application logs, as well as tracking known threats to stay ahead of any newly emerging ones.

8. CONCLUSION

In conclusion, the rising relevance of SQL injection attacks demands robust security measures for database-driven applications. Implementing input validation, parameterized queries, prepared statements, stored procedures, and regular security audits is crucial to mitigate the risks of data theft and system takeover.

Regarding AI algorithms for SQL injection detection and prevention, SVM, CNN, and Naive Bayes each have strengths and weaknesses. For optimal results, the choice depends on specific data and application requirements. CNN algorithms show high accuracy and effectiveness in detection. Remember, the effectiveness of AI algorithms depends on the quality of the training data. Continuous updates and training on evolving threats are essential to ensure long-term effectiveness. Organizations must also understand the algorithms' limitations to make informed decisions about their implementation.

SQL injection attacks pose serious threats to organizations and individuals:

- Security risk: They jeopardize sensitive data, leading to theft or corruption of personal, financial, and business information.
- Widespread impact: All SQL database systems are vulnerable, making them a significant threat to organizations of any size.
- Easy execution: Simple to launch using automated tools, increasing the risk.
- Result in losses: Financial, data, reputational, and legal consequences.[21]
- Often undetected: Attackers can exploit systems without detection.

Preventing SQL injection brings several benefits:

- Data protection: Safeguarding sensitive information in databases.
- System stability: Avoiding crashes and data corruption.[22]
- Improved reputation: Demonstrating commitment to customer data protection.
- Reduced legal and financial risks: Mitigating potential fines and lawsuits.
- Increased efficiency: Minimizing disruptions to business operations.[23]
- In conclusion, a comprehensive security strategy and updated AI algorithms are essential to detect and prevent SQL injection attacks effectively.

In conclusion, a comprehensive security strategy and updated AI algorithms are essential to detect and prevent SQL injection attacks effectively.

REFERENCES

SQL Injection Monitor - Detecting SQL Injection Attacks | SolarWinds. (n.d.). Retrieved June 7, 2023, from https://www.solarwinds.com/security-event-manager/use-cases/sql-injection-attack

(N.d.). Researchgate.net. Retrieved June 7, 2023, from https://www.researchgate.net/publication/362157464_

Hu, J., Zhao, W., & Cui, Y. (2020). A survey on SQL injection attacks, detection, and prevention. Proceedings of the 2020 12th International Conference on Machine Learning and Computing.

What is SQL Injection? Tutorial & Examples. (n.d.). Portswigger.net. Retrieved June 7, 2023, from https://portswigger.net/web-security/sql-injection

(N.d.-b). Sas.com. Retrieved June 7, 2023, from https://www.sas.com/en_us/insights/articles/risk-fraud/data-visualization-crime.html.

(N.d.-c). Researchgate.net. Retrieved June 7, 2023, from https://www.researchgate.net/publication/353025675_SQL_Injection_Attacks_Prevention_System_Technology_Review

Alsahafi, R. (n.d.). SQL injection attacks: Detection and prevention techniques. Ijstr.org. Retrieved June 7, 2023, from https://www.ijstr.org/final-print/jan2019/Sql-Injection-Attacks-Detection-And-Prevention-Techniques.pdf

SQL injection. (n.d.). Malwarebytes. Retrieved June 7, 2023, from https://www.malwarebytes.com/sql-injection January. (2022, January 1). Top 5 predictive analytics models and algorithms. Insight software. https://insightsoftware.com/blog/top-5-predictive-analytics-models-and-algorithms/

What is an SQL injection attack? See examples & learn prevention. (n.d.). Rapid7. Retrieved June 7, 2023, from https://www.rapid7.com/fundamentals/sql-injection-attacks/.

What is an SQL injection? Cheatsheet and examples. (n.d.). Spiceworks. Retrieved June 7, 2023, from https://www.spiceworks.com/it-security/application-security/articles/what-is-sql-injection/

What is SQL Injection? Tutorial & Examples. (n.d.). Portswigger.net. Retrieved June 7, 2023, from https://portswigger.net/web-security/sql-injection

Hasson, E., & Cheng, L. (n.d.). What is SQL injection? Learning Center. Retrieved June 7, 2023, from https://www.imperva.com/learn/application-security/sql-injection-sqli

What is SQL Injection (SQLi) and how to prevent attacks? (n.d.). Acunetix. Retrieved June 7, 2023, from https://www.acunetix.com/websitesecurity/sql-injection

Shakya, A., & Aryal, D. (n.d.). A taxonomy of SQL injection defense techniques. Diva-portal.org. Retrieved June 7, 2023, from http://www.diva-portal.org/smash/get/diva2:830374/FULLTEXT01.pdf

(N.d.). Veracode.com. Retrieved June 7, 2023, from https://www.veracode.com/blog/research/how-prevent-sql-injection-attacks

SQL Injection. (n.d.). Owasp.org. Retrieved June 7, 2023, from https://owasp.org/www-community/attacks/SQL_Injection

Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2022). Detection of SQL injection attack using machine learning techniques: A systematic literature review. Journal of Cybersecurity and Privacy, 2(4), 764–777. https://doi.org/10.3390/jcp2040039

(N.d.-b). Portswigger.net. Retrieved June 7, 2023, from https://portswigger.net/web-security/sql-injection/prevention

(N.d.-c). Dzone.com. Retrieved June 7, 2023, from https://dzone.com/articles/sql-injection-prevention-in-java

(N.d.-d). Acunetix.com. Retrieved June 7, 2023, from https://www.acunetix.com/blog/articles/understanding-preventing-sql-injection-attacks/

(N.d.-e). Sitepoint.com. Retrieved June 7, 2023, from https://www.sitepoint.com/sql-injection-protection-php/

(N.d.-f). Red-gate.com. Retrieved June 7, 2023, from https://www.red-gate.com/simple-talk/sql/t-sql-programming/sql-injection-prevention-parameterized-queries/