Journal of Positive School Psychology http://journalppw.com
2022, Vol. 6, No. 8, 6315-6326

Early Detection And Mitigation Methods For Sql Injection Attacks Using Adaptive Free Algorithm

Mr. N. Aravind Kumar[1*] , B. Sujatha[2]

[1] Ph. D Scholar, Department of Computer Science Engineering, Osmania University, Hyderabad Telangana, India. , Email: aravindkumar.nuvuri@gmail.com,
[2] Department of Computer Science Engineering, Osmania University, Hyderabad Telangana, India.
E-mail: sujata.banothu@gmail.com
**Corresponding Author: - Mr. N. Aravind Kumar*

Abstract—SQLIA (Structured Query Language Injection Attack) is a relatively common web security flaw. The attacker inserts harmful Structured Query Language (SQL) script into an online form's input in order to control answers to information or make unauthorized modifications to it. A successfully hostile SQL injection has substantial economic, reputational, security, and governmental consequences for the afflicted firm. Numerous studies have been conducted on the detecting and preventing of SQL injection attacks. Nevertheless, significant single tools for the both detection and prevention of SQL injection threats still are lacking. We've presented a method for detecting and mitigating SQL injection attacks to use a single tool, which helps software engineers to spot SQL injection security flaws throughout the assessment process. In this work, deep learning approach is proposed for detection of SQL Injection Attacks using Convolutional Neural Networks (CNNs). Then to prevent the SQL Injection Attacks, authentication by means of Asymmetric Encryption is proposed which is known as Optimum Prime Number Selection based Elliptic Curve Cryptography (OPNS-ECC). In this case, genetic algorithm is proposed for the optimum number selection. The proposed methodology is based on parametric searches and authentication of user input. Our findings reveal that the tool is fully accurate and effective when it comes to identifying and mitigating SQL issues.

Keywords—SQL Injection Attacks, CNN Technique, Multiple Features and Detection & Mitigatio.

Introduction

Internet services have now become an integrated component of everyone's daily lives. Conventional key services like finance and commerce are now being supplied on the online network, necessitating the creation of protection for these services. The Internet's accessibility has heightened the danger. It is common to identify severe flaws in a large group's cyber security due of vulnerabilities in its system or software. SQL injection is a common sort of online database server hacking technique. SQL (Structured Query Language) is a database language for adding, deleting, changing, and querying data from the data databases. Therefore long as the government relies on databases, Sql queries are used to interface with it. Injection attacks issues are caused by developers using the method of appending text to generate SQL statements that are given to the database when writing code. As a result, attackers can modify the SQL statement by using SQL words or special characters. The system is attacked as a consequence of the execution. The main motivation is to place too much faith in data given by users, while screening input from the user and neglecting to undertake primary factor in determining verification, in order to fulfill the suspect's goal.

SQL injection disables the verification logic and compromises the database's confidentiality or tries to manipulate it. It aids the intruder in getting access to the database's back end. Whenever a web service cannot provide a proper validation system for the data supplied by the user in the input field, it becomes vulnerable. Surrounding the situation can be used to test for weaknesses in a web

application and can also be used to test for incorrect input queries. Nevertheless, the decrease in service availability caused by denial of service, especially in the case of false positives, is a drawback. Following a thorough investigation, it was discovered that the source of vulnerability is perhaps the programmers or the development pipeline. According to statistics, more than 80% of all current Internet sites are susceptible to SQL injection. The qualities of a good defense mechanism are as follows.

- **Detection:** An SQL injection attempt should be detected and identified by the protection system.
- **Prevention:** The defensive system must have a thorough understanding of SQLInjections and be able to locate them inside the app.

The WebSSARI tool analyses inputs using a preset median filter in order to discover security flaws in the app. This is extremely valuable since it can find stored procedures that are shared by multiple SQLIAs. SQLR, which would be based on randomized query methods, is another option. In the SQL keywords, a strong random integer is inserted. So rather than string manipulations, SQL DOM uses a set of subclasses that are firmly connected to a data structure to generate SQL statements. Improper grammatical forms are the attacks in SQL CHECK, which is predicated on a value injected at both the start and the end of the needs of the user. SQLGAURD presented a temporal architecture of anticipated queries. To identify and mitigate SQL injection security flaws, many frameworks have been used and/or suggested in the research. We'll go over a few of the most prominent ones today. Using word embedding and CNN we present a lightweight solution to preventing SQLIA in this paper. We clean and decoding HTTP requests, and then exploits the Word2vec to construct word embedding's of these decoded characters, train a CNN, and afterwards utilize the classification model for detect fraudulent demands.

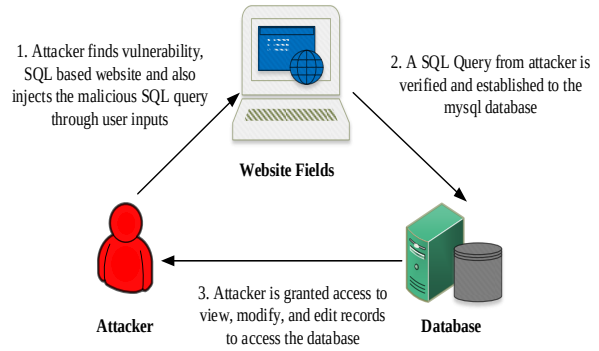


Fig.1. Flow Diagram for SQL Attacks Detection and Prevention

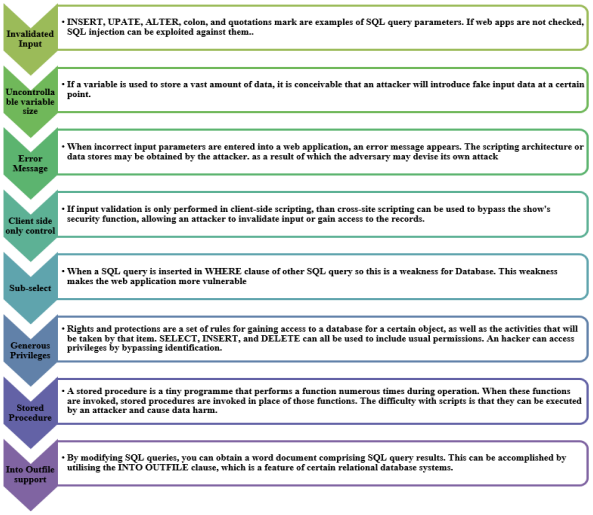


Fig.2. Causes of SQL Injection Attacks

The following is how the paper is organized: The second section examines similar research in this field. In Section 3, we describe our approach in detail, and in Section 4, we show the results of the experiment. Finally, in Section 5, we end the work with a statement on suggestions for future research.

I. LITERATURE SURVEY

Nearly 67 percent of Web sites have been found to unnecessarily expose server information. The percentage of websites that do not secure session cookies is around 50%. In order to communicate critical information to users, around 30% of Websites encrypt their interactions. As either a result, a research on assault mitigation is intriguing. SQL injection is aimed towards interactive Internet services that use databases, and it seeks to attack a weakness in the three-tier architecture's underlying database. Using a methodology influenced by the idea of generic decryption, this study provides a technique for detecting and prevents SQL injection attacks. The

collected findings supported the suggested program's exceptional performance in comparison to current preventative strategies [11].

A SQL Injection Attack (SQLIA) is a form of code injection attack that jeopardises the security, security, and scalability of database servers. The attacker typically uses poorly screened user inputs, such as text fields in web applications, to inject malicious SQL queries into a valid query. Absent sufficient authorization, the attacker can gain access, insert, change, or remove essential information in a database. They explain and categories forms of SQLIA in this review, as well as examine existing detection and prevention approaches for such assaults [12].

Currently, web applications are frequently used. The majority of these web applications are based on money transactions, such as online baking, e-shopping, online pay bills, fund transfers, and so on. Structured Query Language (SQL) and Scripting Language are being used to communicate among web applications and data. These inquiries save sensitive or personal data from a variety of people. As a result, secrecy must be protected from unwanted access. SQL injection attack (SQLIA) is the most prevalent sort of weakness, in which a specially prepared query is used as input to retrieve personal data about other users. Several detection and protection approaches for Known vulnerabilities are given in this study, along with an evaluation of them [13].

In this research, a strategy is proposed that concentrates on request framework identification of SQL injection attacks and query reconstruction. By dynamically rebuilding harmful queries, the suggested architecture can reduce denial of service and boost availability [14].

The aim of this paper is to provide an overview of SQL Injection Attacks (SQLIA) and ways for preventing them. We'll go over all of the recommended models for preventing SQL Injections. Throughout this article, we go over the many types of SQL injection attacks and how to prevent them in online applications. We also go over how to avoid injection threats caused by dynamic Sql query in database configuration files, which are common in e-commerce applications. Along with providing the research outcome, we also discuss future predictions and the future benefit of SQL Injection defenses. SQL Injection, database security, and data structures are some of the key words [15].

Mostly on basis of extensive international and domestic research, we offer a SQL injection detection approach which does not rely on a background rule base by utilizing a natural language processing model and deep learning framework. The strategy can enhance performance and minimize false alarms by allowing the systems to learn the language model characteristics of Sql injections autonomously, eliminating interaction with people and offering partial protection from never-before-seen attacks [16].

This application seeks to protect databases from SQL injection attacks. There is no need to develop this system because it is online. This can be accessible via the internet from any location. To make the information and data safe, this framework employs Injection attacks protection. It will place a special emphasis on employing the AES (Advanced Encryption Standard) technology to encrypt credit card data. Payments are protected by the shop, ensuring everything is safe. The user's credit card information is subsequently saved in a database. The device also uses AES encryption to save user information in an encrypted fashion. The architecture is built to prevent SQL injection attacks that could compromise the database and cause it to malfunction [17].

Although various statically, temporal, and mixed ways to preventing Software vulnerabilities have indeed been proposed, none single approach provides perfect protection or discovery of these attacks. Each year, the CVE repository receives hundreds of reports of open source and commercial software components that are susceptible to SQL injection. We identify distinct existing methodologies in terms of computing cost and security provided in this mapping analysis. We discovered that the majority of current systems promise to provide security based on extremely small or restricted testing. This research deconstructs each recommended method, highlighting its merits and drawbacks, and classifying them according to the technology involved utilised to identify or counteract injections assaults [18].

In today's world, web applications are widely used for different of purposes such as online shopping, online money transfers, e-bill payment, and online

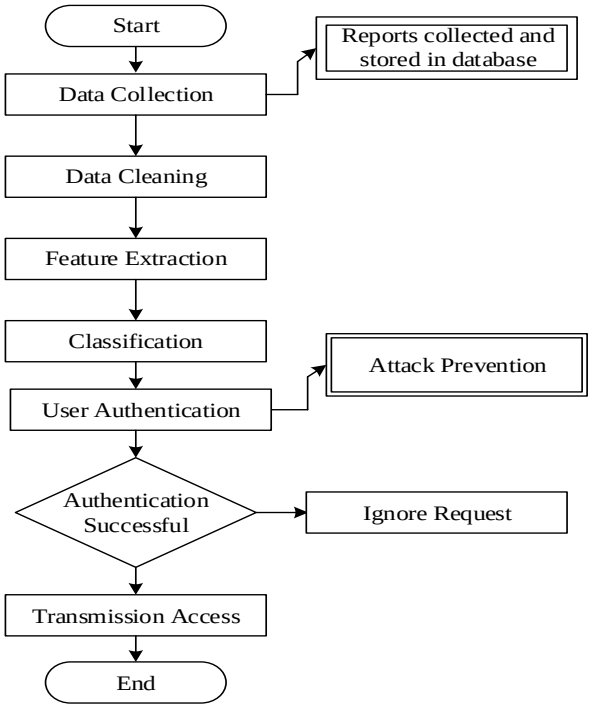
mobile recharges, to name a few. As people become more reliant on these web programmes, the number of attacks against them rises. SQL injection attacks (SQLIA) and Cross-Site Scripting (XSS) are significant web application security issues. SQL injection attack (SQLIA) is the most prevalent sort of vulnerabilities in which a malevolent mind enters its own prepared queries as inputs for obtaining private information about other vulnerable individuals. Various strategies for detecting and preventing SQL injection attacks are given in this study, along with an evaluation of them [19].

Conventional way of identifying SQL injection attacks in online applications include software and hardware-based Web Application Firewalls, programmatic defence techniques such as data filters, error handling, employing parameterized queries, and static and dynamic analysis. We offer a strategy to detecting SQLIA using NLP and Machine Learning in this work. The technique can identify SQLIA with accuracy, recall, and a f1-score of 99.9%, according to experimental data [20].

In SQLIA, an attacker injects data into a query to modify the architecture of the request as planned by the programmer and thus gain access to the information in the database layer. The safety of a web application against SQLIA is critical due to the importance of the information stored. In this research, we offer a new technique for detecting and preventing SQLIAs based on static analysis and runtime verification. User inputs in SQL queries are deleted in this technique, and certain information is captured to make identification easier and more efficient at runtime. Our tests reveal that our suggested methodology is quick, has a high degree of precision, and has near-perfect detection accuracy [21].

II. PROPOSED WORK

Conventional multi - layer perceptron and Web Application Firewalls (WAF) encounter numerous challenges as attack tactics evolve. Because of the advancements in computer resources and the rapid growth of deep learning technology, the focus of this research is to use experimental investigation to choose a computational intelligence framework for detecting whether HTTP queries comprise malicious script for SQL injection attacks, in attempt to optimise detection performance and minimize false alarm rates. Generally, an attacker will wrap the injection attacking code in different ways to get beyond the WAF and filtration mechanism.



As little more than a result, before constructing the deep learning model, the machine learning technique should be cleaned using recursion decoding, following by regular expressions to construct word embedding modeling, and finally fed into the cnn model. We utilised a lightweight model called a convolutional neural network (CNN) to assess the studies in the this study. The model's efficiency is proven by evaluating the dataset, and the training process is demonstrated utilizing Python TensorFlow to evaluate the model's ease of use.The general organization of the testing system is depicted in Figure 3. ECC is a public-key encryption that makes advantage of the elliptic curve's algebraic structure. Cryptography, digital certificates, and pseudo-random generators all use elliptic curves. If compared to other asymmetrical encryption algorithms, it employs little keys. As a result of elliptic curve theory, ECC creates public and private keys. The following is how the cryptography theory is built on the elliptic curve equation:

$$y^2 = x^3 + ax + b \qquad (1)$$

The corresponding elliptic curve for equation 1 can be plotted as shown in fig.3. For that, the following terms are used: E (elliptic curve), P (point on the curve) and n (maximum limit for the prime number).

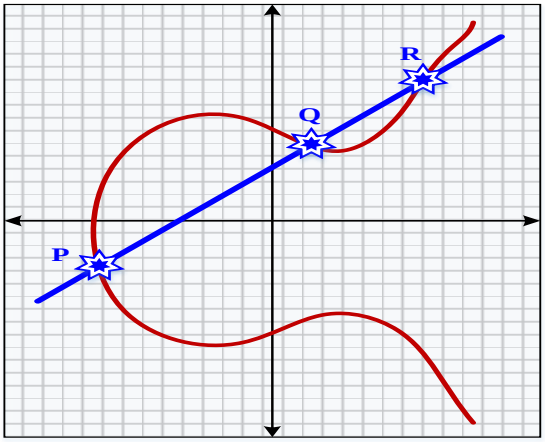


Fig.3. Elliptic curve

(1) Key generation

From the elliptic curve, the public key (Q) is generated as follows,

$Q = Pr * P$ (2)

Here Pr is the random number which resides within the range of n. And P is the point of the elliptic curve. In above equation, Q is the public key and Pr represents the private key.

(2) Encryption

Consider a data D on elliptic curve. Then, a random number k is selected from $[1, (n-1)]$. Upon these parameters, two cipher texts Ct_1 and Ct_2 are generated for d as follows,

$Ct_1 = k * P$ (3)

$Ct_2 = D + k * Q$ (4)

(3) Decryption

For received Ct_1, Ct_2 the original data is extracted as follows,

$D = Ct_2 - Pr * Ct_1$ (5)

In this manner, the ECC algorithm generates public and private keys and performs authentication process for the attacks prevention.

Different embeddings, such as urlencode, querystring, JSON, PHP serialise, base64, and others, may be present in the user's HTTP request data traffic. Prior to training, we must decode all input till it is approved by the final application, referred to as payload. Since an input can decode other packets through some other decryption methods, and all these payloads could be Injection attacks declarations, you can't just exit the loop decrypting once it's been decoded once. Instead, you could perhaps treat these packets as a new data and iteratively decode them again, which is one of the main reasons why SQL injection is difficult to avoid. To lessen the influence of numbers and some extraneous elements on the sample, generalisation is required upon decode to making the pattern's features more evident. The criterion is to split the word after replacing the number with "0" and the URL with http://u.

The Lite module is the significant part in the Lite Net architecture which is integrated in order to provide higher accuracy in determining the data as a fault one. The information provided in the packet header is considered for the prediction of fault and redundant data. The massive incoming data with low interval of time are filtered to detect the fault data. The computational cost in implementing the lite module is reduced by incorporating only small filter size. The filters in the lite module perform both depth wise and standard convolution. The structure of the lite module is provided in the fig.4. The advantage of implementing lite module is the optimal extraction of features from the data for performing fault detection.

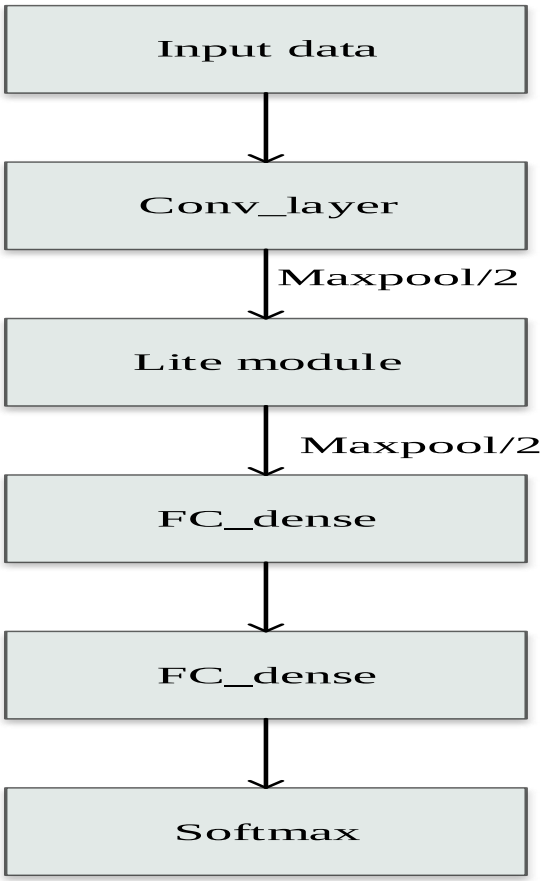


Fig.4. LiteNet architecture

The LiteNet architecture comprised of a standard convolutional layer, lite module, fully connected layer and softmax layer. The softmax layer classifies the data into two classes hence the probability distribution of the output can be computed as,

$$\sum_{i}^{2} p_i = 1 \qquad (7)$$

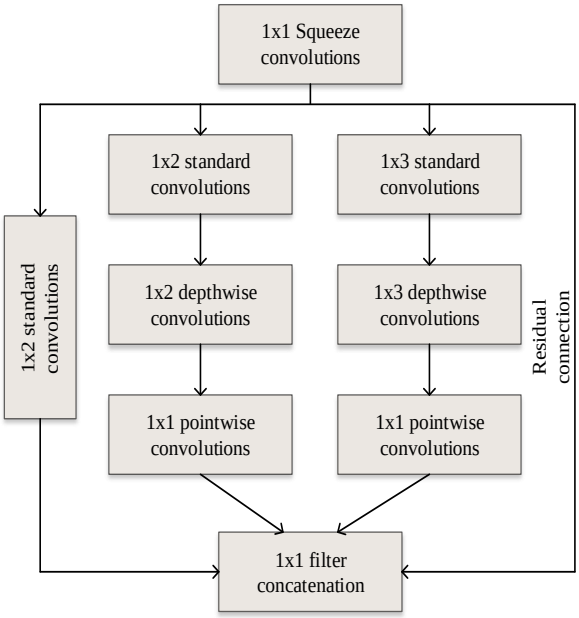


Fig.5. Lite Module

The input provided to the softmax layer from the upper layers for the purpose of fault detection can be formulated as,

$$B_i = \sum_{k} y_k W_{ki} \qquad (8)$$

Where, y is the output obtained from the above layers and W denote the weight of the connection between the softmax layer and the above layers. The detection of fault data by the softmax layer is executed using the hassanat similarity. The similarity between two packets p_a and p_b can be computed as,

$$d(p_a,p_b) = \begin{cases} 1 - \dfrac{1 + min(p_a,p_b)}{1 + max(p_a,p_b)}, min(p_a,p_b) \geq 0 \\ 1 - \dfrac{1 + min(p_a,p_b) + |min(p_a,p_b)|}{1 + max(p_a,p_b) + |min(p_a,p_b)|}, min(p_a,p_b) < 0 \end{cases} \qquad (9)$$

Through this, the detection of redundant and fault data is performed and the detected fault data are pruned which reduces the wastage of storage and complexity in computing the unwanted data. The architecture of the LiteNet model is presented in the fig.5 and the layer information of the model is presented in table 4.2

Table.4.2 Structure of layers in Lite Net architecture

Layers		Kernel size	No. of. filters	Stride
Conv_layer		1 × 5	5	1
Max-pooling		1 × 2	5	2
Lite Module	Squeeze conv_layer	1 × 1	3	1
	Standard Conv_layer	1 × 1	6	1
		1 × 2	6	1
		1 × 3	6	1
	DepthwiseConv_layer	1 × 2	6	1
		1 × 3	6	1
	Pointwise conv_layer	1 × 1	6	1
		1 × 1	6	1
Max-pooling		1 × 2	18	2
FC_dense		-	30	-
FC_dense		-	20	-

The core GA algorithm is based on genetic and evolutionary tendencies, and it mimics the reproductive behaviour of biological populations. The GA uses the survival of the fittest principle in its search phase to select and produce people that are well-suited to their surroundings, including such project requirements or restrictions. As a result, desirable features (features designs) will be involved and stay in the populations genomic composition (set of engineering solutions generated each iteration) above characteristics with stronger unwanted characteristics over a number of cycles (generations). Begin your search in GA with a completely random community of designs that evolve across generations, eliminating the need for a consumer starting place. The GA recommends three main operations for optimisation. The very first operator is the "selection operator," which is based on the survival of the fittest principle. The "crossover operator," which simulates mating in natural populations, will be the next provider. "Mutation" is the final operator, which encourages variability in design demographic features. GA is explored in depth inside the following sections. GA is a bio-inspired method that excels at searching enormous search spaces for the best solutions. The following are the main components of GA:

- encoding technique
  // chromosome, and gene
- initialization procedure
  // creation
- fitness function evaluation
  // environment
- parents selection
  // reproduction
- genetic operators
  //mutation & recombination
- settings of parameters
  // practice & art

The flowchart of the GA is depicted in fig 6. As a result of various elements and supporting nature of global and huge search space based optimal solutions finding operations, GA has obtained various benefits such as (1). GA is very easy to understand, (2). Supports for Multi-Objective Optimization Problem, (3). Good for Noisy Environments, and (4). Easily Distributed in Nature and always results better with respect to the time. The following drawbacks are faced in traditional GA,

- Selecting the basic implementation values such as Representation, Population Size, Mutation Rate, Selection, Deletion Policies, Crossover and Mutation Policies.
- Therefore, the performance of scalability is affected due to the aforementioned factors.

There are many ways to improve the performance of GA for variety of applications. One of the solutions to speed up and make the accurate solution is the Deep learning. Deep learning algorithms are easy to adapt for massive inputs and also alternate or adjust the solutions based on the current iterations. For that substantial history of the previous records are trained in a minimum amount of loss.

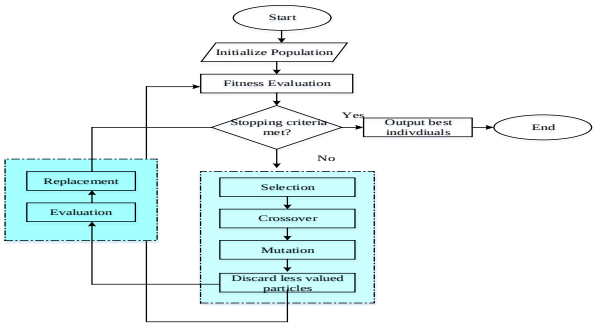


Fig.6. Genetic Algorithm Flow Chart

Pseudocode for GA

Begin
Input - *Numbers*
Initialize population
Compute fitness function for all candidate routes
While *(stopping criteria is met?)*
Output best individual
Else
Rank the middle population
Apply three genetic operators
Perform selection
Perform crossover
Perform mutation
Discard less valued chromosomes
End while
End
Output *– Route Selected*

In GA, the crossover is a Generic Operator the computes the new form of prime number for sending the aggregated packets. The process of crossover is depicted in fig 6. In hidden layer, crossover is applied for all candidates. From that, the set of available numbers are identified. In mathematically, it is represented as follows,

$\Pi\Phi = \{r_1, r_2, .., r_{\mathbb{u}}\}$ if $\mathbb{u} > u(10)$

III. Experimental Results and Discussion

This section discusses about the proposed SA-IDPS approach performance results with the detail experimental evaluation. This section is comprised of four sub-sections including simulation set-up & configuration, definition & formulation of performance metrics, comparative analysis and research highlights. On each section some graphical representations are given. With the obtained results, researchers proved that the proposed approach provides the best results while existing methods or not. The experimental evaluation of the proposed work provides the better results in terms of all performance metrics.

Each split up into two sections: white sampling and black sampling, and each dataset is split into two groups: train and testing dataset. As negative class, there are 25487 instances of SQL injection from the internet, and 24500 samples of a typical HTTP request as positive instances. We chose a data set that includes 4000 SQL injection data and 4000 normal data for testing during data processing.A detailed discussion of simulation results are illustrated as follows .This section illustrates the definition of all performance metrics that are involved to test the proposed performance. Each metric is defined in the following:

A. Accuracy

First of all, elements to compute accuracy are well-defined.

- **True Positive (TP):** It is the value that shows the number of attacks that have been classified correctly as “Intrusion”.
- **False Negative (FN):** It is the value that shows the number of attacks that have been classified as Normal by mistake.
- **False Positive (FP):** It is the value that shows the number of normal packets that have been classified as “Intrusion” by mistake
- **True Negative (TN):** It is the value that shows the number of normal queries that have been classified as “Normal”.

After the basic elements are computed, accuracy is well-defined which can be explained as: Accuracy is the sum of packets that are categorized correctly than total number of queries sent, which is computed by:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (11)$$

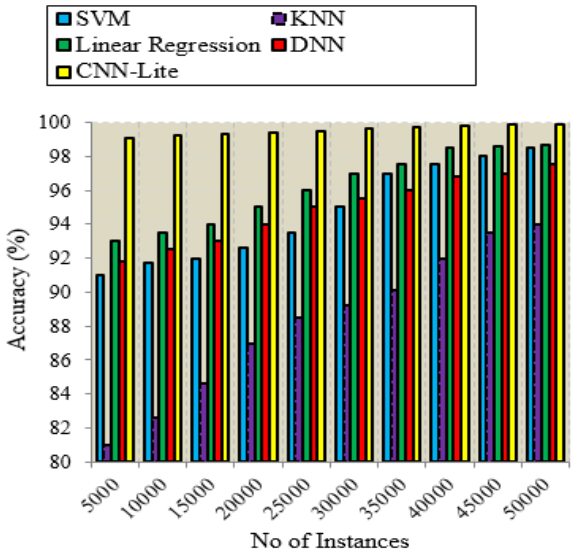


Fig.7. Accuracy vs. Number of Instances

The performance of accuracy for the proposed CNN method and also the other conventional approaches are illustrated in Fig 7. As compared to

the existing methods, the proposed CNN method provides higher classification accuracy. This is relatively higher for the traditional methods to reach this amount of the performance.

B. Attack Detection Rate

With the use of TP attack detection rate is calculated. It is defined by the sum of queries categorized correctly as normal as total number of normal queries are sent. It is computed by:

$$\text{ADR} = \frac{\#\text{ of detected attacks}}{\#\text{ of attacks}} \times 100\% \quad (12)$$

(or)

$$\text{ADR} = \text{TPR} = \frac{\text{TP}}{\text{TP+FN}} \quad (13)$$

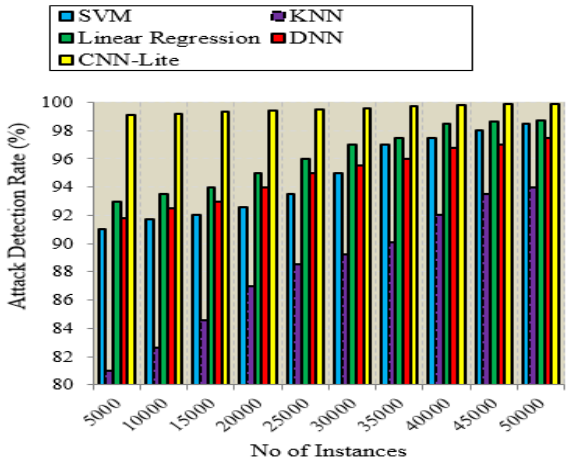


Fig.8. Attack Detection Rate vs. Number of Instances

C. False Positive Rate

It is also similar to FP, which is computed by the total number of queries are classified as anomaly by mistake than number of queries sent. It is computed by:

$$\text{FPR} = \frac{\#\text{ of misclassified processes}}{\#\text{ of normal processes}} \times 100\% \ (14)$$

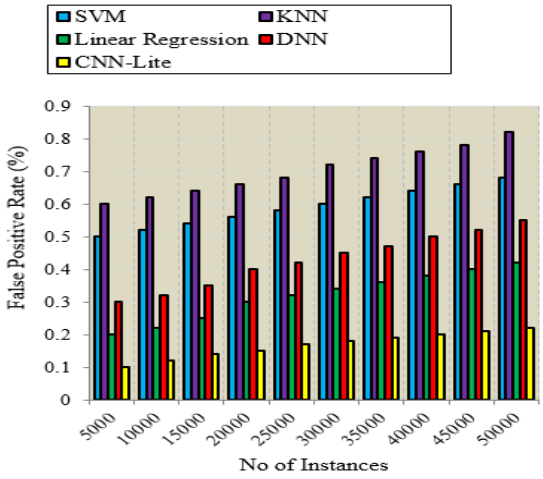


Fig.9. False Positive Rate vs. Number of Instances

D. Detection Delay

It refers to time that required to detect the total number of anomaly queries at time Δt by classifier. In other words, it is the attack detection starting time and ending time.

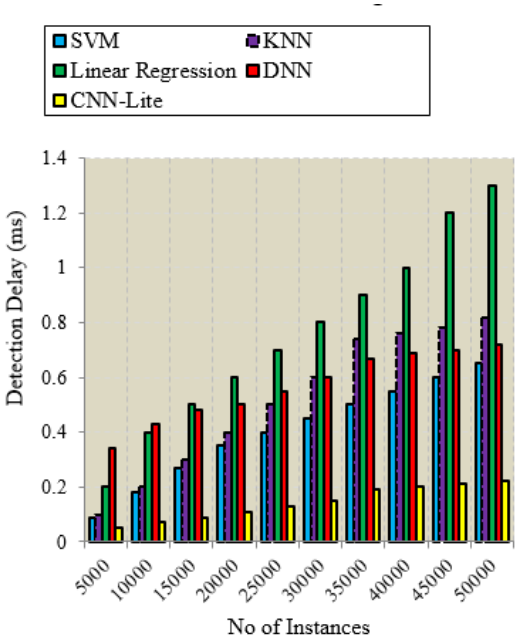


Fig.10.Detection Delay vs. No of Instances

Detection delay is an important factor when design an attack detection scheme because timely detection of attacks may help to avoid a huge network damage or loss by attackers. In this study, delay is considered as an important criterion to show the performance

$$\text{DD} = \text{AD}_{\text{ST}} - \text{AD}_{\text{ET}} (15)$$

The vertical axis shows the detection delay in terms of seconds and the horizontal axis represents different number of attacks. The colored bar shows the existing vs. proposed works. Overall, it can see a clear downward trend in the number of attacks, but by existing works this had upward trend in the same number of attacks. Hence, the proportion of the proposed approach decreased sharply. In addition, scatter plots provides a clear performance representation for the detection delay in a statistical way.

When analyze detection delay statistically for the proposed solution, it resulted 0.2seconds and 0.21 seconds when attackers increase from 40 to 45, respectively. In Figure 10, attacker time interval varied graphical plot is given. Due to attacker participance in a network for different time

intervals, detection is delayed. In addition, it is gradually increase when attack time interval increases.

E. Throughput

In network, packets are delivered via wireless links and also transmitted through either single hop or multi-hop communication. It is measured in bits per second (Bit/s or BPS), which can be measured as data queries per second or per time slot.

Throughput = $\sum_{i=1}^{n}$ NPR / $\sum_{i=1}^{n}$ NPS × Num_H(16)

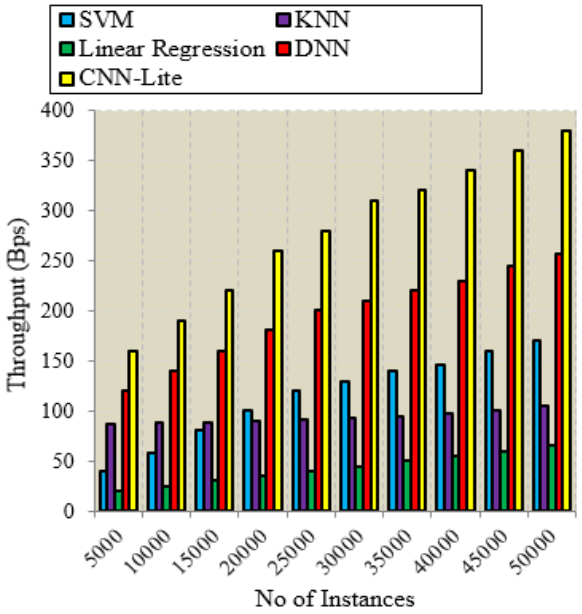


Fig.11. Throughput vs. No of Instances

Where NPR is the Number of Queries Received, NPS is the Number of Queries Sent, and Num_H is the number of hops from the origin to the destination node.

F. Overhead Analysis

In this section, overhead analysis is given. However, overhead must be lesser to show the network has achieved a good performance. Most of the previous works have obtained high performance in terms of accuracy and false alarm rate, but its faces some overhead. One of the big reasons for huge overhead is increased number of samples. Instead of considering the whole trained sets for attacks detection, some specific attributes (features) are used, which reduces overhead in intrusion detection.

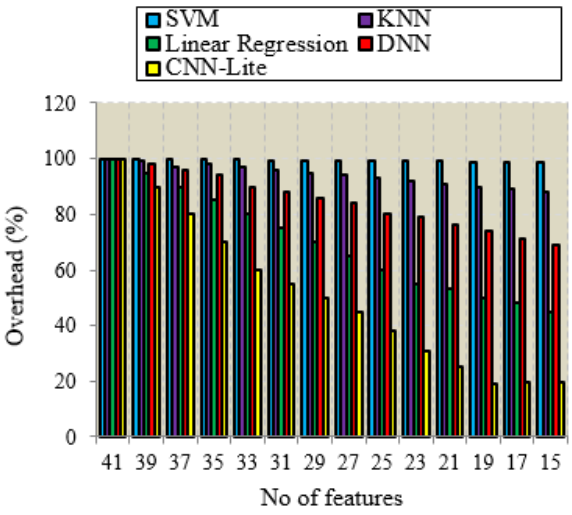


Fig.12. Overhead vs. No of Selected Features

Fig12 represents the overhead with respect to number of selected features. Each work overhead rate is mentioned in this fig 12 for number of features from 41 to 15. The proposed approach requires fewer features that adaptive to reduce the overhead rate. Existing works does not change the overhead when number of selected features change. The proposed approach shows less overhead than existing works as it obtained 19% of overhead for 19 selected features which compared to all existing works.

IV. Conclusion

A popular web security problem is SQLIA (Structured Query Language Injection Attack). In order to control answers to information or make unauthorized changes to it, the attacker inserts a malicious Structured Query Language (SQL) script into the input of an online form. The economic, reputational, security, and governmental ramifications of a successfully hostile SQL injection are significant for the afflicted company. Identifying and mitigating SQL injection attacks has been the subject of numerous studies. Despite this, there are no significant single tools for both detection and protection of SQL injection threats. We've presented a way for identifying and mitigating SQL injection attacks using a single tool, which aids software engineers in spotting SQL injection security problems throughout the evaluation process. Convolutional Neural Networks are used in this paper to provide a deep learning strategy for detecting SQL

Injection Attacks (CNNs). Then, to prevent SQL Injection Attacks, Optimum Prime Number Selection based Elliptic Curve Cryptography is offered as a method of authentication using asymmetric encryption (OPNS-ECC). In this scenario, a genetic method is proposed for determining the best number. The proposed solution is based on parametric searches and user input authentication. Our data show that the tool is completely accurate and effective at detecting and resolving SQL errors. In future to add blockchain or any other decentralized technology for attacks prevention in real-time environment.

References

[1]. Kumar, P.M., &Pateriya, R.K. (2012). A survey on SQL injection attacks, detection and prevention techniques. 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), 1-5.

[2]. 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.

[3]. Nofal, D.E., & Amer, A.A. (2019). SQL Injection Attacks Detection and Prevention Based on Neuro-Fuzzy Technique. AISI.

[4]. Atoum, J.O., &Qaralleh, A.J. (2014). A Hybrid Technique for SQL Injection Attacks Detection and Prevention. International Journal of Database Management Systems, 6, 21-28.

[5]. Alsahafi, R.M. (2019). SQL Injection Attacks: Detection And Prevention Techniques. International Journal of Scientific & Technology Research, 8, 182-185.

[6]. Nofal, D.E., & Amer, A.A. (2020). SQL Injection Attacks Detection and Prevention Based on Neuro—Fuzzy Technique. Studies in Big Data.

[7]. Nasereddin, M., ALKhamaiseh, A., Qasaimeh, M., & Al-Qassas, R.S. (2021). A systematic review of detection and prevention techniques of SQL injection attacks. Information Security Journal: A Global Perspective.

[8]. Hlaing, Z.C., &Khaing, M. (2020). A Detection and Prevention Technique on SQL Injection Attacks. 2020 IEEE Conference on Computer Applications(ICCA), 1-6.

[9]. Dasmohapatra, S., & Priyadarshini, S.B. (2021). A Comprehensive Study on SQL Injection Attacks, Their Mode, Detection and Prevention. Proceedings of Second Doctoral Symposium on Computational Intelligence.

[10]. Ghafarian, A. (2017). A hybrid method for detection and prevention of SQL injection attacks. 2017 Computing Conference, 833-838.

[11]. Archana Devi, R., Hari Siva Rami Reddy, D., Akshay Kumar, T., Sriraj, P., Sankar, P.S., & Harini, N. (2020). Prevention and Detection of SQL Injection Attacks Using Generic Decryption. Lecture Notes in Networks and Systems.

[12]. Yigit, G., &Arnavutoğlu, M. (2017). SQL Injection Attacks Detection & Prevention Techniques. International Journal of Computer Theory and Engineering, 9, 351-356.

[13]. Dehariya, H., Shukla, P.K., &Ahirwar, M.K. (2016). A Survey on Detection and Prevention Techniques of SQL Injection Attacks. International Journal of Computer Applications, 137, 9-15.

[14]. George, T.K., & Jacob, P. (2016). A Proposed Architecture for Query Anomaly Detection and Prevention against SQL Injection Attacks. International Journal of Computer Applications, 137, 11-14.

[15]. Kalsi, T.S., & Kaur, N. (2015). DETECTION AND PREVENTION OF SQL INJECTION ATTACKS USING NOVEL METHOD IN WEB APPLICATIONS.

[16]. Chen, D., Yan, Q., Wu, C., & Zhao, J. (2021). SQL Injection Attack Detection and Prevention Techniques Using Deep Learning. Journal of Physics: Conference Series, 1757.

[17]. Chowdhury, S., Nandi, A., Ahmad, M.S., Jain, A., & Pawar, M.V. (2021). A Comprehensive Survey for Detection and Prevention of SQL Injection. 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 1, 434-437.

[18]. Alenezi, M., Nadeem, M., & Asif, R. (2021). SQL injection attacks countermeasures assessments. Indonesian Journal of Electrical Engineering and Computer Science, 21, 1121-1131.

[19]. Dehariya, H., Shukla, P.K., &Ahirwar, M.K. (2016). A Survey on Detection and Prevention Techniques for SQL Injection Attacks. International Journal of Wireless and Microwave Technologies, 6, 72-79.

[20]. Gogoi, B., Ahmed, T., & Dutta, A. (2021). Defending against SQL Injection Attacks in Web Applications using Machine Learning and Natural Language Processing. 2021 IEEE 18th India Council International Conference (INDICON), 1-6.

[21]. Lashkaripour, Z., &Bafghi, A.G. (2013). A Simple and Fast Technique for Detection and Prevention of SQL Injection Attacks (SQLIAs). International journal of security and its applications, 7, 53-66.