Journal of Theoretical and Applied Information Technology

30th November 2023. Vol.101. No 22
© 2023 Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN INTEGRATED ITERATIVE FEATURE SELECTION AND NAIVE BASED ORIENTED SUPERVISED MACHINE LEARNING TECHNIQUE TO DETECT SQL INJECTION ATTACKS IN IOTS BASED COMMUNICATION

ASIFIQBAL SIRMULLA[1], PRABHAKAR M[2]

[1] Computer Science & Engineering, Reva University Bangalore, Karnataka, India

[2] Computer Science & Engineering, Reva University Bangalore, Karnataka, India

E-mail: [1]r20pcs03@cit.reva.edu.in, [2]prabhakar.m@reva.edu.in

ABSTRACT

IoT is a type of networking model which is composed of several wireless and wired networks that are interconnected through internet-based channels. These devices are widely adopted in various types of applications such as home automation, industries and academic purpose. However, this type of easily accessible connectivity has unlocked several challenging issues where maintaining secure and reliable connectivity is a crucial task. In these attacks, injecting the false queries is considered as one of the most challenging security attacks announced by the OWASP and SQL injection is the common type of injection attack. To deal with this issue, machine learning is considered as promising technique which learns the pattern of historical data and detect the attack by processing the suspicious queries. In this work, we present a supervised learning-based machine learning algorithm which is based on the Naïve Bayes classifier. We present a threefold strategy in proposed scheme, which contains a feature modelling phase where a mutual information based feature dependency model is utilized. Later, an iterative feature selection model is also presented to select features iteratively until the feature selection criteria is obtained. Finally, a naïve Bayes classifier model is presented to classify the selected attributes. The performance of proposed approach achieves the average performance as 0.96, 0.958, 0.97, and 0.979 in terms of Avg. Precision, Avg. Recall, F1-Score, and Accuracy.

Keywords: *SQL Injection Attacks, Security, Naïve Bayes, IoT, Kaggle, SQLIA*

1. INTRODUCTION

The Internet of Things (IoT) is one of the most advanced breakthrough in today's world; it enhances our everyday lives by transforming the biological objects that surround us into a data ecology. Nowadays, the IoT is widely adopted in various real-time and offline applications such as industries, security, transportation, health-care, and home automation etc. and many more. Indeed, IoT and big data are driving today's smart grid advancements, and smart metres are getting smarter by adding more important sensing capabilities and increasing connection [1, 2].

Currently, we've noticed major developments in computing and communication technology over the previous years, resulting in a slew of new smart appliances. The Internet of Things (IoT) consists mostly of computationally constrained devices that expand system and user connection in domain-specific applications. As depicted in Fig.1, recent study shows that by 2025, the number of connected "things" is predicted to increase dramatically, reaching roughly 75.44 billion [3].

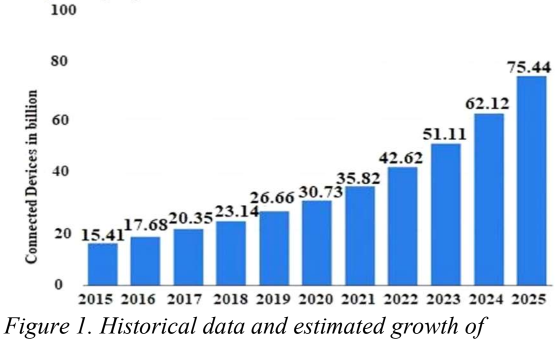

Figure 1. Historical data and estimated growth of connected devices

Generally, the IoT networks consist of sensors and actuator devices which communicate physical systems. This communication involves several daily life objects such as home appliances [4, 5] which are controlled by mobile applications up to largescale infrastructures such as power grids and industrial systems. [6, 7] which are also managed with the help of internet-connected control systems. Due to their vast domain applications in the society, this technology is recognized as critical infrastructure (CIs) by national and international bodies. However, these devices communication over a wireless channel where managing the security becomes the primary concern for the IoT based communities.

While the Internet of Things can provide us with many benefits, it also raises our risk of being exposed to a variety of security and privacy dangers, some of which are novel. Information leakage and denial of service were the most commonly cited security issues prior to the IoT [9]. Theft of information or denial of service aren't the only security concerns with the Internet of Things. These hazards may now be linked to real-life situations, such as physical security. Other worries revolve around privacy. The Internet of Things (IoT) has resulted in a rise in the quantity of personal data provided and exchanged across connected devices. Privacy is a key factor [10], despite the fact that it is not a new demand or unique in this current circumstance.

Security and privacy solutions should be tailored to the features of heterogeneous IoT devices. There is a requirement for security solutions that can provide equal degrees of protection for diverse types of devices without affecting its performance [11, 12]. In IoT based scenarios, each user focus on retrieving the information with the help of internet where the stored data can be in the form of structured or unstructured form with heterogeneous data format. The notion of "Learn Each-Query" can be an effective way to handle the difficulty mentioned above; nevertheless, learning a user's dependent query with no pre-specified notations can be a huge challenge [13]. This challenge may be described as a natural language processing (NLP) problem [14], in which a computer must be trained on known-unknown text properties and make a cognitive conclusion about whether the user or query is benign or malicious [15].

Cyber and physical attacks are the most common types of IoT attacks as depicted in Fig. 2. Cyber-attacks are a type of danger that involves hacking into a wireless network's system in order to manipulate (i.e., steal, delete, change, or destroy) the user's data. Physical attacks, on the other hand, are those that physically harm IoT devices [16]. The attackers do not require a network to assault the system in this case. As a result, physical IoT devices, such as mobile phones, cameras, sensors, routers, and so on, are exposed to these types of attacks, in which the attackers disrupt the service. In general, the attackers attack several levels of the IoT.

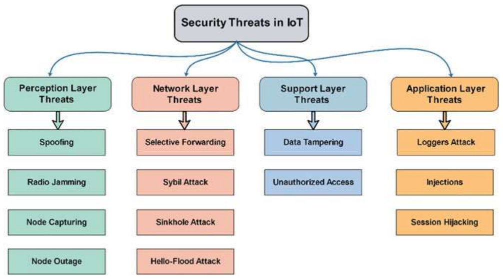

Figure 2. Different types of attacks and security threats in IoT

Cyber-attacks like SQL injection are amongst the most prevalent threats in IoT oriented apps. A SQL injection attack involves inserting or "injecting" a SQL query into the programme via the client's input information. An impactful SQL injection loophole can allow attacker to leak sensitive information from the DB, alter DB data (Insert/Update/Delete), manipulate database administration activities (like shutting down the DB application), restore the data of a provided file on the DBMS file system, as well as, in certain situations, raise unauthorised instructions to the OS.

Users' given data is frequently related with database application security risks. The Structured Query Language (SQL) is a querying, operating, and administration language for database applications. Frequently, user-supplied data is utilised to build the SQL query that queries the DB server. The attackers can alter or transform the data provided by the user, gaining access to the database as a result. As illustrated in Fig. 3, the National Security Agency (NSA) [18] has documented multiple assaults on database systems. The most prevalent assault is a cross-site scripting attack (which accounts for 49.67 percent of the assaults), followed by a SQL injection attack (which accounts for the rest of the attacks) (18.01 percent). The SQL injection attack had also been named the #1 danger to web based systems in 2013 by the Open WEB Application Security Project (OWASP) [19].

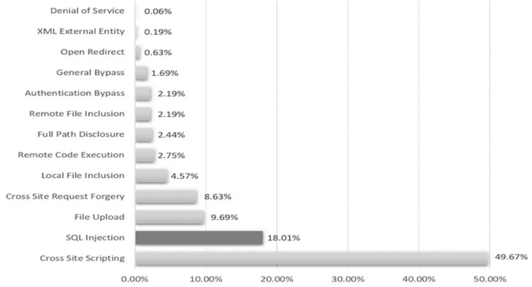

Figure 3. Common attacks on Web-based applications [18]

Several techniques have been introduced to deal with these issues where machine learning has gained huge attraction from research community due to their pattern learning approach which plays an important role to detect different types of attacks based on their historical patterns. Several supervised machine learning based schemes have been introduced in this field such as Naïve Bayes, support vector machine, decision tree and many more.

The traditional machine learning approaches have used machine learning based systems where feature selection, pattern learning and classification are the important tasks. However, these machine learning methods face several challenges such as these methods focus only on static query analysis and fail to focus on dynamic query analysis which results in the poor accuracy. Moreover, these data samples are generated randomly which leads to data imbalance problem. These methods fail to handle the imbalanced data. Similarly, these methods do not consider the transfer learning approaches which can be beneficial in efficient pattern learning. We study about these machine learning techniques in section 2. However, these techniques suffer from issue of attack detection accuracy thus mitigating this is a prime concern. To deal with this issue, we present a novel machine learning scheme with the help of Naïve Bayes classifier. The main contributing of this work are as follows:

- First of all, we present a general model to process the queries in web applications.
- Describe a general model to detect and prevent the SQL attacks
- In order to incorporate the functionality machine learning, we present a feature selection strategy.
- This feature selection uses mutual information calculation to obtain the dependency relation among attributes.
- Later, an iterative process is applied to select these features and these selected features are processed with the help of Naïve Bayes classifier, until the desired outcome is obtained in terms of detection accuracy

Problem Definition

SQL Injection attacks involve maliciously injecting SQL code into input fields of a web application, potentially leading to unauthorized access, data theft, and system compromise. Detecting these attacks in real-time is challenging due to their dynamic and evolving nature. These methods face several challenges such as:

- Complex Attack Patterns: Attackers constantly devise new techniques to bypass security measures, creating complex and varied attack patterns that are difficult to detect using traditional methods.
- Large Data Volumes: Web applications generate massive amounts of data daily, making it challenging to analyze and identify patterns indicative of SQL Injection attacks.
- Imbalanced Data: Genuine user input vastly outweighs malicious input, resulting in imbalanced datasets, which can affect the accuracy of machine learning models.

Dynamic Queries: Web applications often use dynamic SQL queries, making it difficult to differentiate between legitimate and malicious queries.

Based on these research contributions, the main objective of this research can be defined as follows:

- Initially, basic introduction is provided about the work followed by literature review where we studied about existing schemes and identified their drawbacks.
- In next stage, we focus on solving the issues of existing approaches therefore presented a machine learning based solution where feature selection and naïve Bayes classification model is introduced. In this work we have proposed an iterative method for feature selection which uses six different attributes to accomplish the selection process.

Rest of the article is organized in following sections: section II describes the literature review where we study existing machine learning scheme to detect the SQL attacks. However, technique to handle other attacks are also studied. In next section III, we present the proposed solution to improve the classification accuracy with the help of Naïve Bayes classifier. Section IV presents the outcome of proposed approach and comparative analysis to show the robustness of proposed approach. Finally, section V presents the concluding remarks and future scope in this field of SQL injection detection.

2. RELATED WORKS

Previous section has presented brief introduction about IoT devices, its demand, applications, and security related challenges.

Security is considered as one of the most challenging issue for these systems. This section presents a literature review where we have selected articles which are mainly focused on supervised machine learning approach. Moreover, the articles which have worked on feature selection for SQLIA are also studied in this section. Several techniques have been introduced to incorporate the security features in IoT. Static analysis, dynamic analysis, and parameter filtering are the three types of traditional SQL injection detection methods [9]. These attacks are identified by examining the input statements, the static analysis approach finds type and grammatical errors. In this context of IoT security, machine learning is considered as a promising solution to handle the security threats based on the historical patterns and rule based learning. This section presents the brief discussion about existing machine learning techniques which are introduced to detect and classify the SQL injection attacks. In [11] Santhosh et al. considered program interface to detect the SQL injection rather than considering internal characteristics. However, these types of techniques are not able to detect attacks which are in the correct input format. In this direction, Naderi-Afooshteh [27] presented a combined model of stain tracking and black box testing method to prevent the software development cycle from SQL injection. However, the dynamic attack detection was only able to detect the predefined types of attacks which are defined by the application developers and it fails to deal with the attack which has different characteristics from the pre-defined rules.

Hasan et al. [17] reported the severity of SQL injection attacks in web based applications and the level of security threat to the confidential data. Thus, authors introduced developed heuristic machine learning. Along with this, it also includes feature extraction such as comments in the sentence, semicolons, number of commands, abnormal commands, and special keywords etc. In this study, the boosted tree, bagged tree, cubic SVM, linear discriminant and fine Gaussian are used which shows the classification accuracy as 93.8%, 93.3%, 93.7%, 93.7%, and 93.5%, respectively.

Ross et al. [20] used machine learning technique for SQLIA by analysing the multisource type data. This data is collected from host web application and Datiphy appliance node which is situated between webapp and host server. This scheme uses Weka filter StringToVec, correlated feature selection along with genetic search algorithm to obtain the subset of attributes. Later, neural network based classification algorithm is applied to obtained the classification performance.

Tripathy et al. [21] studied about the SQL injection attacks for SaaS applications. Due to increased demand of cloud computing related applications, the utilization of SaaS applications also has increased. However, these applications are purely based on the software applications thus these applications are considered more vulnerable to various attacks. In order to deal with this issue, authors have developed a neural network based machine learning algorithm. Uwagbole et al. [22] introduced a machine learning based approach for SQLIA detection by training the Two-Class Support Vector Machine (TC SVM) and Two-Class Logistic Regression (TC LR) model.

Li et al. [23] discussed that the performance of these systems depends on the feature extraction and selection because increasing multisource data type leads to complexity in feature selection. In order to overcome this issue, authors focused on advanced machine learning technique and presented long short-term memory based automated process of SQL injection attack detection. Moreover, an injection generation method is also presented based on the data transmission channels. This scheme also helps to solve the problem of overfitting which is caused due to insufficient positive samples. Appelt [24] developed an automated process to detect the security holes in the Web application firewalls. This scheme uses a combination of machine learning and evolutionary algorithm.

Gandhi et al. [25] considered two different advanced machine learning concepts which are CNN and BiLSTM. These techniques are based on the concept of deep learning. However, training complexity of this model increases its timing cost. Gu et al. [26] reported that existing web application firewalls schemes of SQL injection detection directly block the suspicious queries without identifying whether the query is harmful or not. To overcome this issue, authors introduced a novel approach called DIAVA which sends proactive warnings regarding the attacks. In [28] Lu et al introduced a new approach called as synBERT for SQL injection attack detection. This process in based on the semantic learning process that explicitly embeds the sentence level semantic information into an embedding vector.

In [29] Falor el al studied various machine learning algorithm such as Naïve Bayes, Decision trees, Support Vector Machine, and K-nearest neighbour and measured their performance for SQL attack detection process. However, the outcome of deep learning outperformed when compared with these methods. Alarfaj et al. [30] reported that traditional methods are able to detect known types of attacks thus fail in real-time scenario. To overcome the issues of existing schemes, authors introduced probabilistic neural network approach to detect these attacks. The feature extraction process is accomplished based on tokenizing and regular expression by using Chi-Square test analysis.

Alghawazi et al. [31] used autoencoder mechanism for attack detection. This model is a combination of autoencoder and RNN. The autoencoder is consist of encoder and decoder module whereas the RNN models comprise of LSTM and dense layer. Nagabhooshanam et al. [32] developed attack detection mechanism for medical data by using Neural Network approach. However, the convergence related issues of NN are addressed by incorporating genetic algorithm based optimization strategy. Sheth et al. [33] suggested to incorporated Markov Decision process in SQLIA and adopted Q learning mechanism to minimize the SQLIA threats. Stiawan et al. [34] focused on SQL and cross-site scripting (XSS) attack detection by using machine learning approach. This model is developed by combining the LSTM and PCA module. Al Wahaibi et al. [35] introduced SQIRL approach which is based on deep reinforcement learning. This mechanism uses multiple worker agents and grey-box feedback.

3. PROPOSED MODEL

As mentioned in previous section, supervised machine learning schemes are widely adopted in various classification problems. This section presents the proposed solution for classification of SQL injection attacks by using Naïve Bayes classification algorithms. Prior to that, we present a brief discussion on working of SQL attacks. Generally, the web applications contain a three-tier architecture which includes presentation tier, CGI tier and database tier, as depicted in below given figure 4. The presentation tier is used to receive the user inputs, the CGI tier is known as server script process which is present between presentation and database tier in the application. Similarly, the database tier is used to store the predefined rules and user entries. In a general process, when user gives the data as input in the form of ID and password to the presentation layer, this data is received with the help of GET function and POST method is used to pass these inputs to database for further process. Below given figure illustrates the process of normal and abnormal queries under the SQLIA.

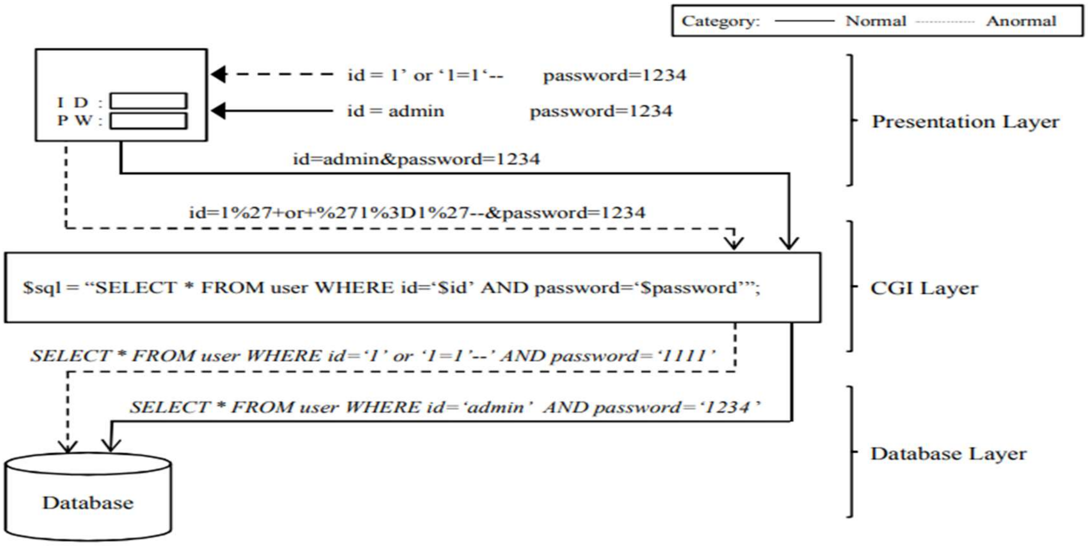
Category: Normal Anormal
I D :
P W :
id = 1' or '1=1'--
password=1234
id = admin
password=1234
Presentation Layer
id=admin&password=1234
id=1%27+or+%271%3D1%27--&password=1234
$sql = "SELECT * FROM user WHERE id='$id' AND password='$password'";
CGI Layer
SELECT * FROM user WHERE id='1' or '1=1'--' AND password='1111'
SELECT * FROM user WHERE id='admin' AND password='1234'
Database
Database Layer

Figure 4. Normal And Abnormal Query Processing In Web Application

Similarly, malicious users may enter the ID in the query form as 1' or '1=1'. In CGI module this query becomes SELECT * FROM user WHERE id = '1' or '1=1'— 'AND password ='1111'. At this stage the, the query or sentence after — becomes a comment because the malicious user has modified the query and it became the input as '1-1' which is always true. This statement bypasses the authentication step. To handle this type of issue, we incorporate an architecture which uses a machine learning based verification technique to process all types of queries.

3.1 Incorporating Feature Extraction for SQLIA

The proposed approach is based on machine learning strategy where analysis of attributes or features of input data plays an important role and it has impact on accuracy. There are several attributes in SQL databases which are generally considered for SQL injection attack detection. Generally, these characters are known as dangerous characters, and dangerous tokens which includes punctuations, special characters. Based on these attributes, we divide all type of attributes into 7attributes which are presented in below given table 1.

Table.1. Attributes And Their Description

Attribute Representation	Description
f_1	Special tokens and their repetitions are marked as dangerous tokens such as drop, rename, create, insert, delete, update etc.
f_2	Special characters and their repetitions are marked as dangerous characters such as $(--,\vert\vert,\backslash,*,@@,/**/)$
f_3	Punctuations such as $\left(*,;,,<>,\{,\},@,\&,+,-,?,\%,,/\right)$
f_4	Frequency tokens and their repetitions are such as table, update, check, create, revoke, convert, deny, concat, group by, join
f_5	Checking the spaces count or frequency in the statement
f_6	Finding the presence of characters or statements in query which always results in true value as outcome such as "@ = @"

Thus, in our case, the query which is not matching with the aforementioned table is considered as the normal query. The other features can be extended as the table is scalable. In order to perform the feature selection, we consider Leave-One-Out Cross Validation (LOOCV) method on the input attribute data which is denoted as $X = \{x_1, x_2, \dots x_D\}$ where X is the complete attribute set, x denotes the value of attribute for corresponding index, and D denotes the dimensionality. In this work, we use an iterative attribute selection which considers x_1 and one other remaining attribute from the attribute set. With this process, it generates the pair of attributes as $X_1, X_2, \dots, X_M$ in M steps. At this stage, we use mutual information measurement to measure the dependency between two attributes. In this context, we consider an attribute F_i and target attribute T. The mutual information between attributes is computed as follows:

$$I(F_i, T) = \int_{f_i}\int_{t} P(f_i, t) \log \frac{P(f_i, t)}{P(f_i)P(t)} dt df_i \qquad (1)$$

If the value of MI is 0, the variables are independent of one another and carry no information about one another, which demonstrates that the attributes are not relevant. Higher MI values suggest that there is more knowledge on the target and hence that the target is more relevant. These features are further used by a simple search strategy which initializes with empty feature subset and adds next features from the list which are having the higher value of mutual information corresponding to current and target attributes. This process of adding the new attributes and updating attribute list is performed until the predefined number of attributes are reached or the performance doesn't improve further.

3.2 Architecture of Detection Engine and its Components

Before presenting the machine learning model, we present the HTTP protocol model along with an attack detection architecture. Generally, the input HTTP stream is divided into two tables such as $HTTP_{request}$ and $HTTP_{response}$. There exists request Hooks for each table which contains security rules declared for each request and response from the table.

- HTTP request: HTTP protocol contains the human readable text in the form of ASCII text. This text contains the header which consists of the request information from the client or browser or the corresponding response from the server. Generally, this HTTP request initializes with the GET or POST methods. The header of the input request provides various information about client, content, connection etc.

3.3 Attack Prevention System

- Dissecting the input URL: in first phase, the user inputs the query and the input URL is processed by the dissector based on the security rules which are defied by user in in the hook. This HTTP URL is further processed and URL strings are extracted and passed to the classifier.
- $IsAuthentic$: the classifier processes the input query and tries to analyses its

keyword. Based on it pattern learning process, the classifier generates the outcome whether the query is authentic or it resembles with attack.

- $IsSQLIA$: the query which is classified as attack in previous phase is further processed by the next module to identify whether the requested query is a type of SQLIA or not. For this purpose, it uses the rules mentioned in HTTP URL hooks. This has two possibilities as: rules are matched and no rules are matched. Here, considering a minimum threshold of rule matching to assign the query as "not authentic" is an important task.
- $Matched$: at this stage, detection engine uses pattern matching algorithm to analyse the specious URL contents and if one of content is matched with the SQLIA rules under the category of attack, then the requested URL is rejected. Similarly, there is another condition where no rules are matched with the rules which are mentioned in attacked category in the HTTP hooks. Thus, it shows $NoRuleMatched$ and the request is classified as authentic request.

The overall process of request process, security rule processing, verifying its legitimacy, and attack classification etc. are presented in below given Fig. 5.

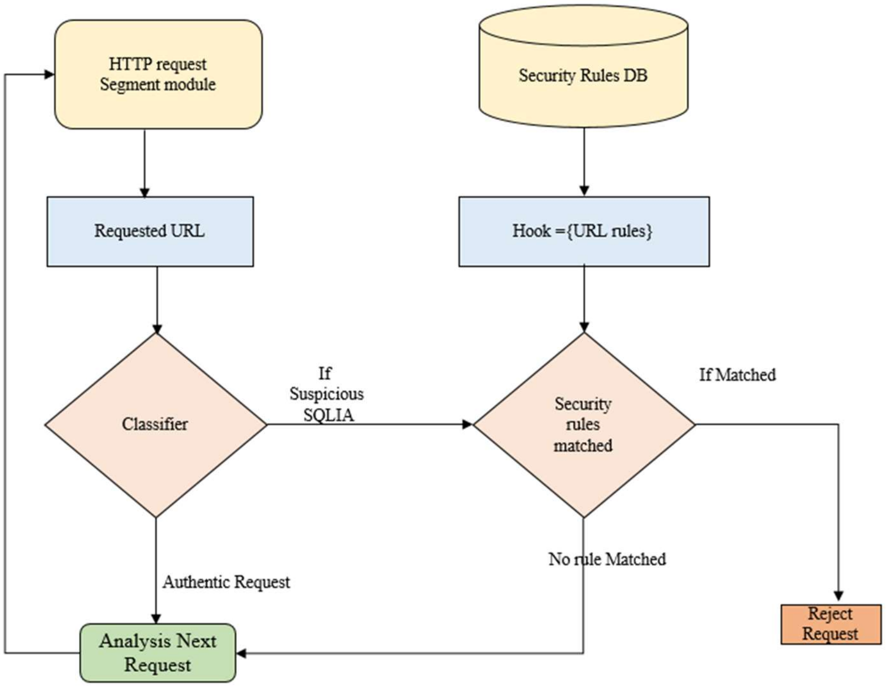

Figure 5. Overview Of Attacks Detection System

3.4 Naïve Bayes classification (NBC) algorithm

In this work, we have adopted a Naïve Bayes based classification algorithm because it allows understanding and alleviating the prediction outcomes in a simpler way. Several studies also have reported that the outcome of NBC algorithm are competitive with respect to the other classifiers. The NBC model is simple to construct and it can handle the large datasets. Moreover, it is a multiclass classification problem which is based on the Bayes theorem which assumes the conditional independence between pairs of input variables.

According to NBC, a set of independent variables can be represented as $X = \{x_1, x_2, \ldots., x_n\}$ which contains two or more classes. The posterior probability of each possible class is denoted as $C = \{c_1, c_2, \ldots, c_m\}$. In this work, we have attack description as attribute and the label corresponds to its class. The basic NBC based on Bayes formula can be expressed as:

$$P(C|X) = \frac{P(C)P(X|C)}{P(X)} \quad (2)$$

Here X denotes the attributes in the data, C denotes the class, $P(C|X)$ probability of even C for occurrence of X, $P(X|C)$ denotes the probability of even X for occurrence of C, $P(C)$ denotes the probability of event C and $P(X)$ denotes the probability of event X. By substituting the values of X, Eq. (2) can be rewritten as:

$$\mathrm{P(C|x_1,x_2,..,x_n)}=\frac{\mathrm{P(C)P(x_1,x_2,...,x_n|C)}}{\mathrm{P(x_1,x_2,...,x_n)}} \quad (3)$$

Further, this expression can be elaborated as follows:

$$\begin{array}{l} P(C|x_1,x_2,..,x_n) \\ = \begin{array}{c} P(C)P(x_1,x_2,\ldots,x_n|C) \\ P(C)P(x_1|C)P(x_2\ldots x_n|C,x_1) \\ P(C)P(x_1|C)P(x_2|C,x_1)(x_3\ldots x_n|C,x_1,x_2) \\ P(C)P(x_1|C)P(x_2|C,x_1)P(x_3|C,x_1,x_2)\ldots P(x_n|C,x_1,x_2,..,x_{n-1}) \end{array} \end{array} \quad (4$$

This expression can be rewritten as:

$$P(C|x_1,x_2,..,x_n) = P(C)\prod_{i=1}^{n} P(x_i|C) \quad (5)$$

The main aim of Naïve Bayes classifier is to maximize the probability value of each class. This process of maximizing the class probability can be expressed as:

$$\begin{aligned} H_{MAP} &= \arg\max P(C|x_1,x_2,\ldots,x_n) \\ &= \arg\max P(C)\prod_{i=1}^{n} P(x_i|C) \end{aligned} \quad (6)$$

3.5 Problem Statement

This section presents the proposed ant colony optimization based strategy for optimal next hop selection to minimize the energy consumption. Ant colony optimization is a probabilistic technique which is widely adopted to deal with computational problems. This optimization approach solves these problems by finding the optimal paths. Artificial agents are used in this approach which adopt the behaviour of real ants where pheromone based communication strategy is used for communication. Below given figure depicts a basic process of ant colony optimization.

Where τ_{ij} denotes the pheromone value, α is the influence on pheromone value on the considered path, β is the influence value on heuristic factor of ant for path selection.

4. RESULTS AND ANALYSIS

This section presents the description of outcome of proposed approach along with the comparative analysis where outcome of proposed approach is compared with other classifiers. The proposed approach is implemented by using Python tool running on windows platform. This system has a capacity of 8GB RAM, 4GB NVIDIA graphic processor and 1TB of storage space.

4.1 Dataset Description

In this work, we have used a publically available SQL injection attack dataset from Kaggle's website. This dataset contains total 34,048 entries into two columns. The first column denotes the sentence which need to be analysed to detect as normal query or SQL injection attack whereas second column represents the corresponding labels for the query (normal or SQL attack). In this data, label 0 denotes the normal sentence and label 1 denotes the sentence as SQL injection attack. This dataset contains 11,781 negative samples and 22,305 positive samples.

4.2 Confusion Matrix

In order to measure the performance, we consider the confusion matrix analysis. First of all, we have obtained the confusion matrix which is given the measurement of correct and incorrect classification as given in table 1.

Table 1 Confusion Matrix

Actual class	Predicted class	
	Normal	Attack
Normal	True Positive	False Negative
Attack	False Positive	True Negative

True positive: it shows that the classifier correctly predicts the positive class from the given test set.

True Negative: it shows that the classifier model correctly predicts the negative class from the given test set.

The true negative and true positive values show the accuracy of classifier. However, these categories should match the actual values of TP and TN.

False positive: denotes the classifier model incorrectly predicts the positive class.

False negative: denotes that the classifier mistakenly predicted the negative class.

This confusion matrix aids us in calculating the suggested approach's total accuracy, precision, specificity, sensitivity, and F-measure. Accuracy is a unit of measurement for the rate of correct categorization, and it is represented by the letter Acc. It is calculated by dividing the overall number of correct predictions by the total number of correct predictions. It can be written as:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

Similarly, the Recall performance also can be computed with the help of true negative and false positive values. This can be computed as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{TN}} \quad (8)$$

The precision of the suggested technique is then computed. It's calculated by dividing the number of True Positives by the number of (True and False) positives.

$$\text{P} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (9)$$

Lastly, the F-measure, and that is the average of accuracy and sensitivity performance, is calculated. It's computed like:

$$F = \frac{2 * P * Sensitivity}{P +} \quad (10)$$

Based on these parameters, we compute the performance of proposed approach and compared with other supervised classifiers.

4.3 Performance Measurement

Here, we demonstrate the outcome of proposed naïve Bayes classifier and compare the obtained performance with other classifiers. During training phase, we consider 70% of data for training and 30%. Based on this configuration, we obtain the following confusion matrix using proposed classifier. Below given table 2 shows the obtained confusion matrix by using proposed approach which contains 874 and 359 instances as correctly classified.

Table.2. Obtained confusion matrix by using proposed classifier

Actual class	Predicted class	
	Normal	Attack
Normal	874	22
Attack	5	359

Further, we compute the precision, recall, F1-score for each class. The obtained values are presented in below given Fig. 6.

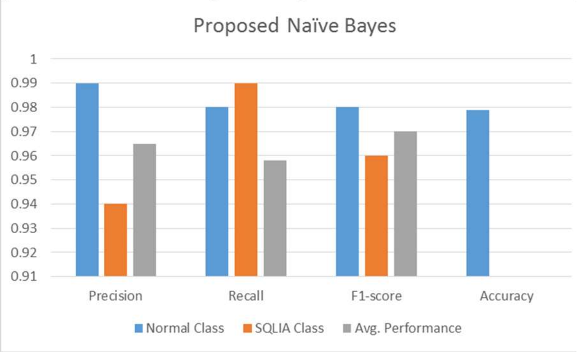

Figure 6. Statistical performance for proposed Naïve Bayes classifier

This comparative analysis shows that the proposed approach achieves the average accuracy of 97.9% for this dataset. Moreover, the average precision, recall and F1-score are obtained as 96.5%, 95.8%, and 97%, respectively. In next experiment, we measure the performance of Naive Bayes classifier for multivariate Bernoulli model. For this experimental also, we obtain the confusion matrix and other statistical performance parameters. The obtained performance values are illustrated in table 3 and Fig. 7.

Table.3. Obtained confusion matrix by using Bernoulli NBC

Actual class	Predicted class	
	Normal	Attack
Normal	825	71
Attack	0	364

Similarly, the statistical performance for this classifier is presented in Fig.7.

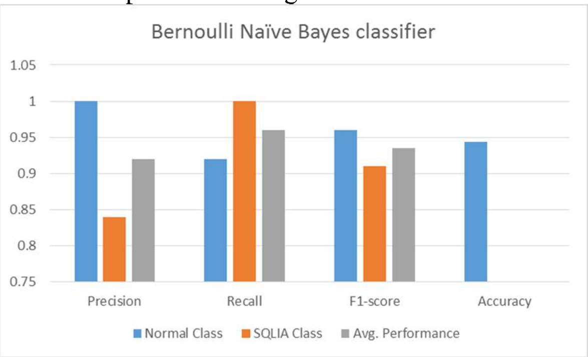

Figure 7. Statistical performance for Bernoulli Naïve Bayes classifier

The average accuracy for this classifier is obtained as 94.36% and other performance parameters such as Precision, Recall, and F1-score values are obtained as 92%, 96%, and 93.5%, respectively.

Further, the performance is measured based on complement NBC where rather than calculating the probability of an item belonging to certain class, we generally compute the probability of the item belonging to all classes. The obtained confusion matrix for complement NBC is presented in below given table 4.

Table.4. Obtained confusion matrix by using Complement NBC

Actual class	Predicted class	
	Normal	Attack
Normal	857	39
Attack	78	286

Based on this confusion matrix, we measure the performance of this Complement NBC. The obtained performance for each class and average performance is presented in Fig. 8.

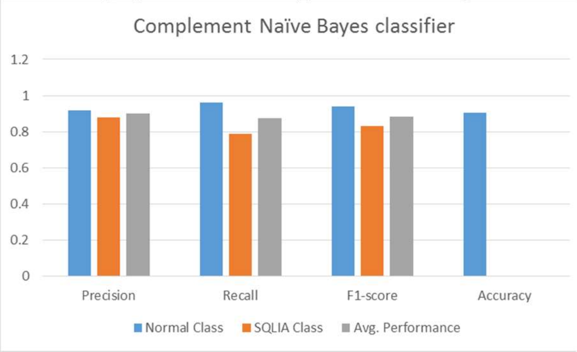

Figure 8. Statistical performance for complement Naïve Bayes classifier

Finally, we measure the performance for Multinomial NBC and retrieve the confusion matrix, classification accuracy and other statistical performance measurement parameters. The obtained outcome is presented in below given table 5 and Fig. 9.

Table.5. Obtained confusion matrix by using Multinomial NBC

Actual class	Predicted class	
	Normal	Attack
Normal	874	22
Attack	91	273

Based on this confusion matrix, we measure other performance related parameters as given in fig. 9.

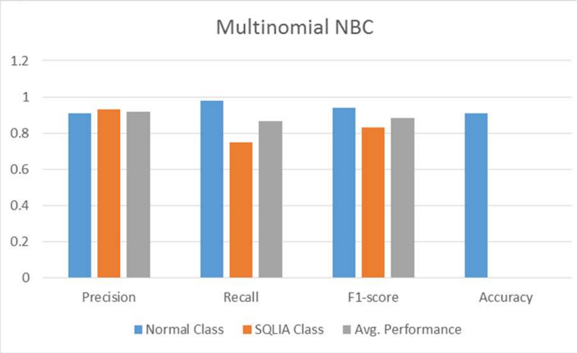

Figure 9. Statistical performance for Multinomial NBC

In this experiment, we obtained the average accuracy as 91.03% and other parameter values such as Precision, Recall, and F1-score are obtained as 0.92, 0.865, and 0.885, respectively.

Based on obtained average precision, recall, F1-score and accuracy, we present a comparative analysis to show the robustness of proposed approach. Below given Fig. 10 illustrates the comparative analysis for aforementioned parameters.

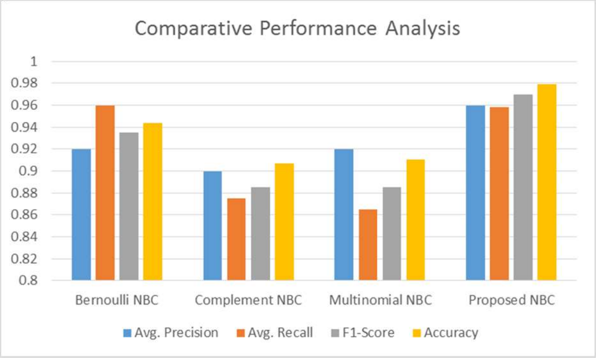

Figure 10.comparative analysis in terms of average precision, recall, f1-socre and accuracy

The comparative analysis shows that the proposed approach achieves the accuracy as 97.9% which is higher than the other machine learning techniques. The proposed feature extraction and selection process is validated by using different types of Naïve Bayes classifiers such as Bernoulli's NBC, complement NBC, and Multinomial NBC. The existing Bernoulli's NBC obtained improved recall performance but proposed model is able to achieve better performance in terms of precision, F1-score and accuracy.

5. CONCLUSION

In this work, we have focused on analysing the IoT enabled automation systems and identified that security is one of the most challenging issue in this field. The IoT devices are more vulnerable to injection type of threats where SQL injection is the top most security threat. The traditional algorithms are not suitable to handle the huge data diversity and fail to detect attacks which are not present in the predefined rules. To overcome the security related challenges of IoTs, we present a machine learning based scheme which uses Naïve Bayes classification technique. It is combined with an iterative feature selection model to select features iteratively until the feature selection criteria is obtained. In order to validate the robustness of proposed model, we tested its performance on publically available dataset and presented a comparative study. The obtained comparative analysis shows that the proposed approach achieves the average accuracy of 97.9% for this dataset. Moreover, the average precision, recall and F1-score are obtained as 96.5%, 95.8%, and 97%, respectively.

REFERENCES:

[1] Hassan, W. H. (2019). Current research on Internet of Things (IoT) security: A survey. Computer networks, 148, 283-294.

[2] Centenaro, M., Costa, C. E., Granelli, F., Sacchi, C., & Vangelista, L. (2021). A survey on technologies, standards and open challenges in satellite Iot. IEEE Communications Surveys & Tutorials, 23(3), 1693-1720.

[3] Al Wahaibi, S., Foley, M., & Maffeis, S. (2023). {SQIRL}:{Grey-Box} Detection of {SQL} Injection Vulnerabilities Using Reinforcement Learning. In 32nd USENIX Security Symposium (USENIX Security 23) (pp. 6097-6114).

[4] Kolanur, C. B., Banakar, R. M., & Rajneesh, G. (2021, July). Design of IoT based Platform Development for Smart Home Appliances Control. In Journal of Physics: Conference Series (Vol. 1969, No. 1, p. 012052). IOP Publishing.

[5] Srinivas, P., Das, M. S., & Latha, Y. M. (2021). Future Smart Home Appliances Using IoT. In Innovations in Computer Science and Engineering (pp. 143-151). Springer, Singapore.

[6] Mehmood, M. Y., Oad, A., Abrar, M., Munir, H. M., Hasan, S. F., Muqeet, H., & Golilarz, N. A. (2021). Edge computing for IoT-enabled smart grid. Security and Communication Networks, 2021.

[7] Sun, D., Li, W., Yao, X., Liu, H., Chai, J., Xie, K., ... & Feng, L. (2021). Research on IoT architecture and application scheme for smart grid. In Proceedings of the 9th International Conference on Computer Engineering and Networks (pp. 921-928). Springer, Singapore.

[8] Stellios, I., Kotzanikolaou, P., Psarakis, M., Alcaraz, C., & Lopez, J. (2018). A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services. IEEE Communications Surveys & Tutorials, 20(4), 3453-3495.

[9] Aversano, L., Bernardi, M. L., Cimitile, M., & Pecori, R. (2021). A systematic review on Deep Learning approaches for IoT security. Computer Science Review, 40, 100389.

[10] Patnaik, R., Padhy, N., & Raju, K. S. (2021). A systematic survey on IoT security issues, vulnerability and open challenges. In Intelligent System Design (pp. 723-730). Springer, Singapore.

[11] Gu, X., & Zhang, Z. (2021). Iot security and new trends of solutions. In Introduction to Internet of Things in Management Science and Operations Research (pp. 55-76). Springer, Cham.

[12] Ahmad, I., Niazy, M. S., Ziar, R. A., & Khan, S. (2021). Survey on IoT: security threats and applications. Journal of Robotics and Control (JRC), 2(1), 42-46.

[13] Gowtham, M., & Pramod, H. B. (2021). Semantic Query-Featured Ensemble Learning Model for SQL-Injection Attack Detection in IoT-Ecosystems. IEEE Transactions on Reliability.

[14] Verma, K. (2022). Modeling Digital Healthcare Services Using NLP and IoT in Smart Cities. In Smart Cities and Machine Learning in Urban Health (pp. 138-155). IGI Global.

[15] Cui, L., Yang, S., Chen, F., Ming, Z., Lu, N., & Qin, J. (2018). A survey on application of machine learning for Internet of Things. International Journal of Machine Learning and Cybernetics, 9(8), 1399-1417.

[16] Shah, Y., & Sengupta, S. (2020, October). A survey on Classification of Cyber-attacks on IoT and IIoT devices. In 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) (pp. 0406-0413). IEEE.

[17] Hasan, M., Balbahaith, Z., & Tarique, M. (2019, November). Detection of SQL injection attacks: A machine learning approach. In 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA) (pp. 1-6). IEEE.

[18] National Security Agency, "Defending Against the Exploitation of SQL Vulnerabilities to Compromise a Network", available at https://www.iad.gov/iad/library/ia-guidance/tech-briefs/defendingagainsttheexploitation- of-sql-vulnerabilities-to.cf.

[19] Mark Churphy, "The Open WEB Application Security Project", available at https://www.owasp.org/SQL_Injection .

[20] Ross, K., Moh, M., Moh, T. S., & Yao, J. (2018, March). Multi-source data analysis and evaluation of machine learning techniques for SQL injection detection. In Proceedings of the ACMSE 2018 Conference (pp. 1-8).

[21] Tripathy, D., Gohil, R., & Halabi, T. (2020, May). Detecting SQL injection attacks in cloud SaaS using machine learning. In 2020 IEEE 6th Intl Conference on Big Data Security on Cloud

(BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS) (pp. 145-150). IEEE.

[22] Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2017, September). An applied pattern-driven corpus to predictive analytics in mitigating SQL injection attack. In 2017 Seventh International Conference on Emerging Security Technologies (EST) (pp. 12-17). IEEE.

[23] Li, Q., Wang, F., Wang, J., & Li, W. (2019). LSTM-based SQL injection detection method for intelligent transportation system. IEEE Transactions on Vehicular Technology, 68(5), 4182-4191.

[24] Appelt, D., Nguyen, C. D., Panichella, A., & Briand, L. C. (2018). A machine-learning-driven evolutionary approach for testing web application firewalls. IEEE Transactions on Reliability, 67(3), 733-757.

[25] Gandhi, N., Patel, J., Sisodiya, R., Doshi, N., & Mishra, S. (2021, March). A CNN-BiLSTM based Approach for Detection of SQL Injection Attacks. In 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE) (pp. 378-383). IEEE.

[26] Gu, H., Zhang, J., Liu, T., Hu, M., Zhou, J., Wei, T., & Chen, M. (2019). DIAVA: a traffic-based framework for detection of SQL injection attacks and vulnerability analysis of leaked data. IEEE Transactions on Reliability, 69(1), 188-202.

[27] A. Naderi-Afooshteh, A. Nguyen-Tuong, M. Bagheri-Marzijarani, J. D. Hiser, and J. W. Davidson, ‘‘Joza: Hybrid taint inference for defeating Web application SQL injection attacks,’’ in Proc. DSN, Jun. 2015, pp. 172–183

[28] Lu, D., Fei, J., & Liu, L. (2023). A Semantic Learning-Based SQL Injection Attack Detection Technology. Electronics, 12(6), 1344.

[29] Falor, A., Hirani, M., Vedant, H., Mehta, P., & Krishnan, D. (2022). A deep learning approach for detection of SQL injection attacks using convolutional neural networks. In Proceedings of Data Analytics and Management: ICDAM 2021, Volume 2 (pp. 293-304). Springer Singapore.

[30] Alarfaj, F. K., & Khan, N. A. (2023). Enhancing the Performance of SQL Injection Attack Detection through Probabilistic Neural Networks. Applied Sciences, 13(7), 4365.

[31] Alghawazi, M., Alghazzawi, D., & Alarifi, S. (2023). Deep Learning Architecture for Detecting SQL Injection Attacks Based on RNN Autoencoder Model. Mathematics, 11(15), 3286.

[32] Nagabhooshanam, N., Murthy, C. R., & CosioBorda, R. F. (2023). Neural network based single index evaluation for SQL injection attack detection in health care data. Measurement: Sensors, 27, 100779.

[33] Sheth, T., Anap, J., Patel, H., Singh, N., & Ramya, R. B. (2023, May). Detection of SQL Injection Attacks by giving apriori to Q-Learning Agents. In 2023 IEEE IAS Global Conference on Emerging Technologies (GlobConET) (pp. 1-6). IEEE.

[34] Stiawan, D., Bardadi, A., Afifah, N., Melinda, L., Heryanto, A., Septian, T. W., ... & Budiarto, R. (2023). An Improved LSTM-PCA Ensemble Classifier for SQL Injection and XSS Attack Detection. Computer Systems Science & Engineering, 46(2).