International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Software As Service Attack Detection and Prevention for Deceitful QR code

[1]Manushree Sahay, [2]Dr. Sandeep Vanjale, [3]Prof. Madhavi Mane

Submitted: 06/09/2023 **Revised**: 22/10/2023 **Accepted**: 05/11/2023

Abstract: The economic benefits and cons of using SaaS (software as a service) are not without debate. The security risks associated with SaaS prevent some consumers and service providers from using it. Using examples including software-defined networking, cloud computing, mobile cloud computing, and the Internet of Things, this article highlights the flexibility and use of SaaS in a number of contexts. Data security, application security, and SaaS deployment security are only some of the SaaS security challenges that will be investigated next. After that, options on how to make a SaaS platform more secure are presented, including how they may work together. The most severe security hole in the SaaS program is the SQL injection attack. The loss of private or crucial information might arise from this. (e.g., financial, personal). Both material (such as data) and intangible (such as reputation) assets might be jeopardized by these types of attacks on a company or organization's sensitive information. The goal of this study is to see whether it is possible to use machine learning methods to identify SQL injections in applications. To prepare the classifiers used in the evaluation procedures, both malicious and safe payloads were used. When given a payload, they can determine whether it contains malicious code. The purpose of this research is to identify malicious actions in a cloud-based SaaS setting. The latest studies on the practicality and safety of QR codes are thoroughly analyzed in the anti-phishing recommendations for this method, known as a secure QR code. We found the most common use cases and the most common attack vectors to exploit them. We did a massive literature search to accomplish this. The most often reported kind of fraud using QR codes as an attack vector is social engineering, sometimes known as phishing. QR codes on cellphones have exploded in popularity in assembly lines for automobiles.

Keyword: *Security analysis, Access control, Emotion interaction, identity authentication, social robot.*

1. Introduction

Storage in the cloud Offers protection against vendor lock-in and other issues that might arise while storing data with many CSPs in the cloud. Third-party cloud storage service provider CSB handles the connection between one or more CSPs and cloud customers through software as a service (SaaS). The cloud is a relatively new idea, and cloud-based storage makes it simple for users to store their data online, where it can be accessed by anyone with an internet connection. However, cloud computing provides a restoration capability that lets clients revert to a previously saved state in the event of an attack or other catastrophic failure. A computer disaster is a simple way to get rid of viruses and other threats to your machine. Intruders may be educated and targeted for remote VM startup and shutdown during brief windows of opportunity. This is a fantastic defensive weapon. The hypervisor's VM-based operation makes it manageable to restrict malicious code. To that end, VM infrastructure will harden itself in the same way a network of physical servers would.

Collision attack: A collision attack is when an adversary takes advantage of such a collision to break the hash's stated security measures. The malevolent user (i.e. untrusted user) launches a collision attack when they supply incorrect or deceptive credentials for a cloud service. When we talk about a malevolent user, we're referring to someone who has never signed up for or utilized the cloud service in question. By requiring users to create unique passwords at registration, this project eliminates the possibility of a collision attack. This is why we verify the user's identity before allowing access with a password. If the individual providing the password is already a registered user, their feedback is accepted and shown to the new user in the form of cloud consumption.

Password login: It is said that two messages, D1 and D2, have a collision if and only if H(D1) = H(D2). It is

[1]*Bharati Vidyapeeth (Deemed to be University) College of engineering Pune.*
[2]*Bharati Vidyapeeth (Deemed to be University) College of Engineering,Pune-43.*
[3]*Assistant Professor, Dept. of Computer Engineering, Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune*
[1]*manushreesahay018@gmail.com,* [2]*sbvanjale@bvucoep.edu.in* , [3]*mmmane@bvucoep.edu.in*

computationally challenging to find a collision using the Brute Force approach, yet a collision can be discovered in every case. There are two distinct sorts of collisions: strong and weak.

SQL injection: Another common attack vector is SQL injection, which happens when SQL queries are executed with user-supplied content included in the query string. When attempting to decode a QR code, a data injection attack may be launched against the reader's structured objects. An attacker may craft a malicious QR code that inserts potentially harmful strings anywhere in the target user's data structures.

SQL Injection Attack: Out-of-band, Boolean-based, time-based, and error-based models were all considered. SQL injection attacks are only one example among many. As an example of an error-based SQLi vulnerability, the following queries may be used by attackers to get information like table names and content. Attackers can test for web application vulnerabilities to SQLi by inserting a condition into an SQL query. If the page loads as usual it indicates that the page is prone to attack. The following query is an example:

Id=1+AND+1=1

U_Id = 105 OR 1=1;

Username '' or ""=""

User_Id = 105; DROP TABLE Suppliers;

QR code: There are varying sizes of QR codes with various sections designated for particular applications. Version 2 of QR codes is used in the sections that follow (Figure 1)

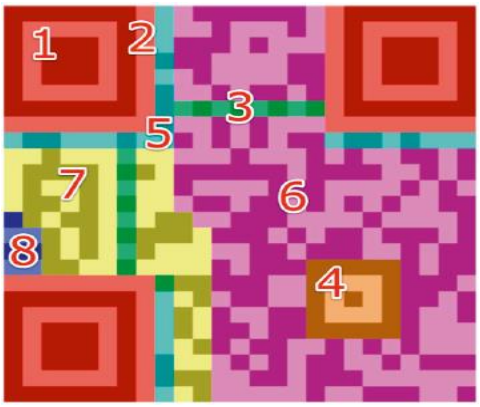

Fig. 1.1 Structure of QR code Version 2

Finder Pattern (1): The finder pattern consists of three similar structures spread across the QR code with the exception of the lower right corner. Each design is predicated on a 33-by-3 matrix of black modules surrounded by white modules, which are in turn surrounded by black modules. The Finder Patterns allow the decoder software to recognize the QR code and establish the correct orientation.

Separators (2): The Finder Patterns are easier to see because of the white separators, which are one pixel wide and serve to set them apart from the actual data.

Timing Pattern (3): The Timing Pattern alternates between black and white modules, which allows the decoder software to determine the width of a single module.

Alignment Patterns (4): The decoder software may make up for mild visual imperfections with the help of Alignment Patterns. QR codes of version 1 do not include Alignment Patterns. More and more Alignment Patterns are introduced as the code base expands.

Format Information (5): The QR code's error correction level and the selected masking pattern are recorded in the 15-bit Formation Information segment immediately after the separators.

Data (6): In the data portion, information is transformed into a bit stream and stored in 8-bit chunks (called code words).

Error Correction (7): Code words in the error correction segment are 8 bits long, much like data codes.

Remainder Bits (8): Since error correcting bits cannot be evenly split into 8-bit code words, this portion is made up of zeros.

2. Literature Survey

According to [1] a method for identifying SQLIA that uses a vocabulary based on reserved terms to spot suspicious query tokens. The first phase generates a lexicon, and the second tokenizes the input query statement, with each string token being matched to terms in a predetermined lexicon, all with the goal of avoiding SQL injection attacks. Successful experiments have been conducted using several approaches for detecting and preventing SQL injection attacks. Since the declaration of tautology ('a'='a') has been appended to this query, it can never be false. The SQL parser will ignore anything after the double dash "--" since it is a remark.

According to [2] Since the system exists online, further development is unnecessary. Thanks to the internet, you can get to this from anywhere. The database and its contents are protected from harm by means of SQL injection in this framework. The use of AES (Advanced Encryption Standard) for the protection of credit card information will be highlighted. The store provides payment protection to ensure a risk-free transaction. Credit card information provided by users is entered into a database. Information entered by the user is encrypted using AES before being stored on the device. The framework is built to prevent SQL injection attacks, which may be used to compromise and execute a database.

According to [3] in order to identify fraudulent QR Codes, a novel model has been provided. During the QR Code validation process, the suggested model will be able to identify phishing and harmful URLs and block the user from verifying them. The creation of this program will aid in protecting consumers from being duped by malicious QR Codes. This work is geared for people like students and workers who often use QR code scanners and want to feel safe that their personal data won't be compromised in the process. This detection model's strength is in its ability to identify the aberrant and harmful connections that are covertly included in QR codes and exploited as vectors for attacks. Before being introduced, this QR code phishing detection tool underwent extensive development and testing. Protocols including HTTP, HTTPs, FTP, SFTP, RTSP, Telnet, NNTP, and Gopher have been utilized to ensure that malicious and phishing pages are spotted during scanning.

According to [4] a categorization of the latest methods for detecting and avoiding the SQL injection attack. We organize the many origins, objectives, and varieties of attacks. In addition, we go through and categorize the most current and pivotal suggested solutions for counteracting this assault, with a focus on those that make use of ontologies and machine learning. The basics of the SQL injection attack were laid forth. We explore and detail the many origins, motivations, and targets of attacks. The different countermeasures against SQLI attacks were compared in a table. Solutions based on ontology and machine learning, among others, are discussed and debated.

According to [5] employing a natural language processing model and deep learning framework, and informed by extensive local and international research, we provide a technique for detecting SQL injections that does not depend on a background rule base. Reduced reliance on human involvement and protection against theoretical 0 day assaults are two benefits of this approach, which also enables the computer to automatically learn the language model elements of SQL injection attacks.

According to [6] the method of encoding a secret QR code into separate shares as a visual secret sharing mechanism. When compared to previous methods, the proposed scheme's shares are legitimate QR codes that can be deciphered with the use of a modern QR code reader, making them more difficult to track down for malicious actors. Existing methods of transferring data are vulnerable to hacking. Based on this assumption, the strategy for (k, n) get to structures may be analyzed by looking at the frequency of the (k, k) sharing event on each k-member subset given a certain connection. Additionally, the eligible shares are XORed, which helps retrieve the secret message. This task is easily accomplished with the help of smartphones or other QR scanning devices. Research into optimum partitioning algorithms and the use of hashing techniques to compare the original message with the shared message are contributions.

According to [7] when an attacker uses a rapid response code to try to trick their victims into giving over sensitive information, such as a password, personal identification number, username, or online banking data, they are committing a phishing assault. Since more and more individuals are using their phones not only for talking but also for making purchases, this kind of attack is

becoming more common. Quick response code is widely used by retailers and customers since it is simple to create and use. This is a semantic attack that takes advantage of users' ignorance of the potential dangers lurking inside a QR code's short form. This research analyzed the methods previously utilized to identify this semantic phishing assault. Each research technique's advantages, disadvantages, and unfilled areas of study are highlighted.

According to [8] the primary goal is to identify dangerous URLs hidden behind QR codes. Ninety thousand URLs, half of which are safe and half of which are harmful, were collected from a variety of sources, and their lexical features were extracted. Fuzzy logic and a multilayer perceptron artificial neural network (MLP-ANN) were used as computational intelligence models and contrasted. The best classifier for spotting malicious URLs was found to be an MLP-ANN, and on the basis of this discovery, a proactive, secure, real-time computational intelligence barcode scanner implementation (BarCI) against harmful QR code links was developed..

According to [9] a comprehensive look at how machine learning may be used to identify SQL injection attacks and the many kinds of assaults that can be carried out. We conclude the essay with several suggestions for moving forward, including potential developments in defenses against SQL injection vulnerability. Data manipulation and schema modifications in databases are made possible through the Structured Query Language (SQL). SQL injection vulnerabilities might exist on this site if it makes inappropriate use of it. Due to the ease and low barrier to entry of SQL injection attacks, they may do significant damage to a website for little investment.

According to [10] existing parsing and tagging methods are used to aid in the extraction of SQL injection patterns. The SQL injection detection approach is based on a simple and effective pattern-based Neural Network model. This study aims to uncover a universal pattern in both valid and malicious SQL queries by analyzing their WHERE clauses with the help of natural language tagging methods.

3. Proposed System Details

The system for searching keywords using similarity-based techniques on encrypted data also classification the document based on weight and query. The system is designed for SaaS (Software-as-a-service) Attack detection and prevention of malicious activities in the cloud environment**.** We offer a safe system of information exchange amongst members of the element community. The customers may confidently obtain their public keys from the assembly chief once we provide a protected way for key distribution through secure communication channels. The data owner, the user, the cloud server, and the attacker, who is an untrusted entity, make up the three components of our suggested system. In this step, the data owner uses a cryptographic technique to upload the data file to the cloud server. The data owner receives a successful notice after the data has been saved to the database. The data owner has unrestricted access to the data file he chooses to share, allowing him to provide that file's contents to any group manager. Then it will immediately be available to everyone in the group. Anytime, wherever, all group members have access to every file stored in the cloud. Even if he tries to use SQL injection queries to construct a collision attack, our system will block it. The total method dramatically enhances both system effectiveness and safety.

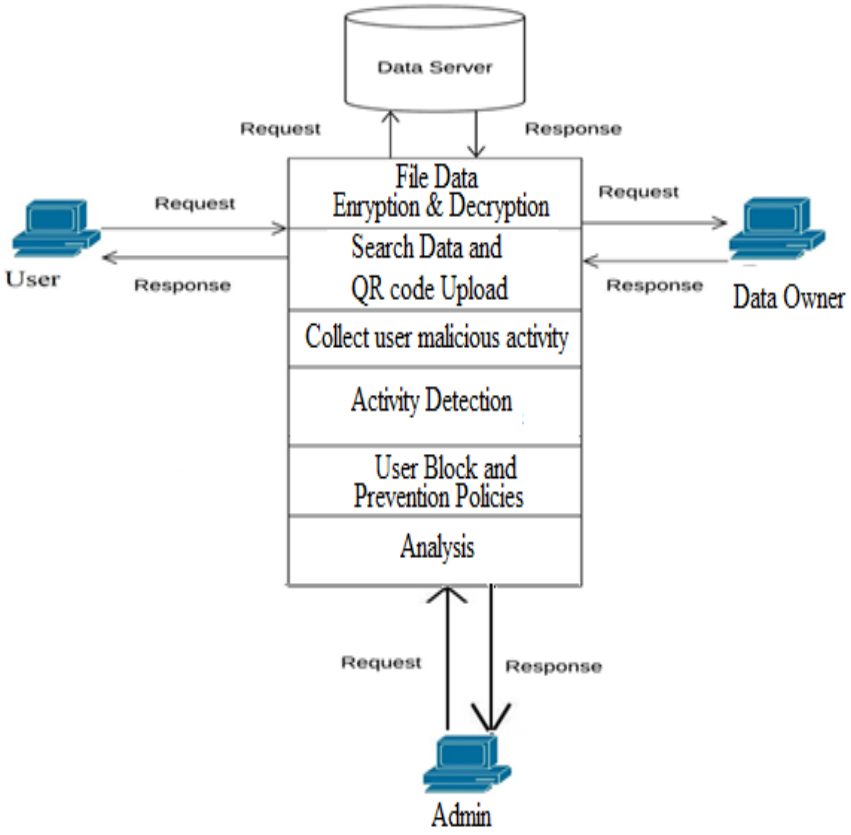

Fig 2: Proposed System architecture

Login and Registration Module: Once a person registers in this section, they'll be able to log in to the rest of the system. With cloud computing, any authorized user may access the system from any location at any time.

Data Uploading: Once the data owner uploads the file, the first step begins. Keys are encrypted using the PBEWithMD5AndDES technique and sent to the EC2 Cloud in the same module. The data's owner will save a copy of the file online and may provide a trusted friend or colleague access to it. By entering the credentials that were supplied to his registered email id, he would be able to view the submitted file. When a user uploads a file, an algorithm will check it against previously submitted files to see whether they are comparable. When several users submit the same file, the system only registers the initial owner's copy and allows the other users to view it as "friend users," preventing the server from storing multiple copies of the same data.

Encryption and Decryption: Decryption is the process of making an encrypted communication legible and understandable. Encryption is the process that transforms a secure message (plaintext) into a cipher (chipper text) that can no longer be deciphered. Encryption and decryption are processes that need a cryptographic key or keys. A cryptosystem is a system that can transform data from its unencrypted form into an encrypted one, and vice versa. Based on the nature of the keys used in the encrypting and decrypting processes, cryptography may be classified as either symmetric key cryptography or asymmetric key cryptography.

Data Sharing: At this point, the data owner has completed data sharing and may send any file to any member of the cloud group. In order to access the file that was shared with him by the data owner, the friend user must go through the appropriate login procedure and use the correct credentials that were sent to him for that file.

Access Control and revocation: When there is no restriction on who may see a shared file, everyone can access it. Data owners have the ability to revoke a user's access to a file in the context of revocation. The shared files posted by numerous users will be removed from the

system when the maximum retention period set by those users has elapsed.

File request and download: user can give the download request to cloud server, at the same Data Owner at verification has done.

Cloud Storage Service Provider (CSP)

Database is provided by Cloud Storage Service Provider. The data owner has the option of storing anything. Users of CSP are also given the option of creating their own custom database structure. CSP will allot space for the user instance according on their specifications.

4. Algorithm Design

Similarity (Machine learning) Algorithm:

Input: Normative training set Testset using normalized data from Train_Data The threshold Th_values was specified by Test_Data.

Output: Output set using the parameters {Predicted_class, weight_Score}

Step 1: The code below may be used to read every test result from Test_Data [] and check training rules. The data is then normalized and altered to meet the needs of the algorithms.

$$\text{test\_Feature(data)} = \sum_{m=1}^{n} (.\ \text{Attribute\_Set[A[m] …… A[n]} \leftarrow Test\_Data)$$

Step 2: choose the features from the test's extracted attributes set. With the code below, create a feature map using the data as features.

$$\text{Test\_Feature\_List [t…..…n]} = \sum_{x=1}^{n} (t) \leftarrow \text{test\_Feature(x)}$$

Test_Feature_List [x] contains the features that were chosen for further inspection.

Step 3: Next, the whole training dataset must be reviewed in order to formulate the rule that will be used to classify all test data.

$$\text{train\_Feature\_List(data)} = \sum_{m=1}^{n} (.\ \text{Attribute\_Set[A[m] …… A[n]} \leftarrow Train\_Data)$$

Step 4: Create the training dataset from the input dataset using the function below.

$$\text{Train\_Feature\_list [t…..…n]} = \sum_{x=1}^{n} (t) \leftarrow \text{train\_Feature(x)}$$

The rules list Train_Feature_list[t] produces the feature vector used to construct the hidden layer. That uses the train data to assess all test cases.

Step 5: The similarity weight is determined once the feature map has been generated.

$$\text{Gen\_weight} = \text{CalcSimilarity (Test\_Feature\_List} \,||\, \sum_{i=1}^{n} \text{Train\_Feature\_List[i])}$$

Step 6: Compare your present weight to your goal weight.

$$\text{if(Gen\_weight} >= \text{qTh)}$$

Step 7: $\text{Out\_List.add (trainF.class, weight)}$

Step 8: Go to step 1 and continue when $\text{Test\_Data} ==$ null

Step 9: ReturnOut_List.

Algorithm 2: Password Based Encryption With DES

Generation key for encryption and decryption

1.Char Rk[]=char.random[Rand()];

2.String –data key-data=(String)char Rk[];

3.Return –data *key _data*

Encryption Process

Input: User input text data as *plain-text,* key *key _data*

Output: Generated cipher *Enc-string*

1. Generate a new instance of ABE with MD5 and DES
2. Change the mode to encrypt using cipher instance
3. Plain-byte[] ← plain-text.toByteArray[]
4. encData[]=apply cipher-data method on (*Plain-byte[] , key _data*)
5. Enc-string=Encode 64BaseEncoder on(*encData[]*)
6. Return *Enc-String*

Decryption Process

Input: cipher text *Enc-String*, key *key _data*

Output: Plain text *p_data*

Step 1: Decryption should be done using the key ***key _data***

Step 2: Set Decryption mode with cipher instance.

Step 3: Byte_Data[] ← Decode 64BaseDecoder(***Enc-String***)

Step 4: byte utf [] =apply decipher method on (***Byte_Data[], key _data***)

Step 5: ***p_data*** =convert into string class (utf)

Step 6: return ***p_data***

Algorithms 3: Role Based Access Control Algorithms:

Input: Attribute Email-ID, File Data and File key_data.

Output: Rule set as policies or signatures.

Step 1: Prepare the data string S_list [].

Step 2: Prepare a=0, k=0, User Email-ID

Step 3: Read Filedata and filekey

a← {filekey list [i . . . . . n]}

k← {Email-ID List [i . . . . . n]}

Step 3: for each (read a to S_list)

If (key_data. Equals (a) && User Email-ID.Equals (k))

Then User File Share information show

Else

Then User File Not Share information show

End for

Step 4: End Procedure

5. Results and Discussion

Java 1.8 and NetBeans 8.0, both open sources, are used for developing the suggested framework, which runs on an open-source VM based environment. The hardware configuration for the Intel 2.7 GHz processor uses 12 gigabytes of RAM.

Table 1: Performance comparison of proposed and existing methods

Attack Type	No. of input values	Correct Detection	Accuracy
SQL Injection	140	138	98.57
Collusion	150	149	99.33

The Table 1 describes detection accuracy for 2 different attacks such as SQL injection and Collision attacks. The number of malicious inputs and correctly detected by algorithm are mentioned using attribute 2 and 3.

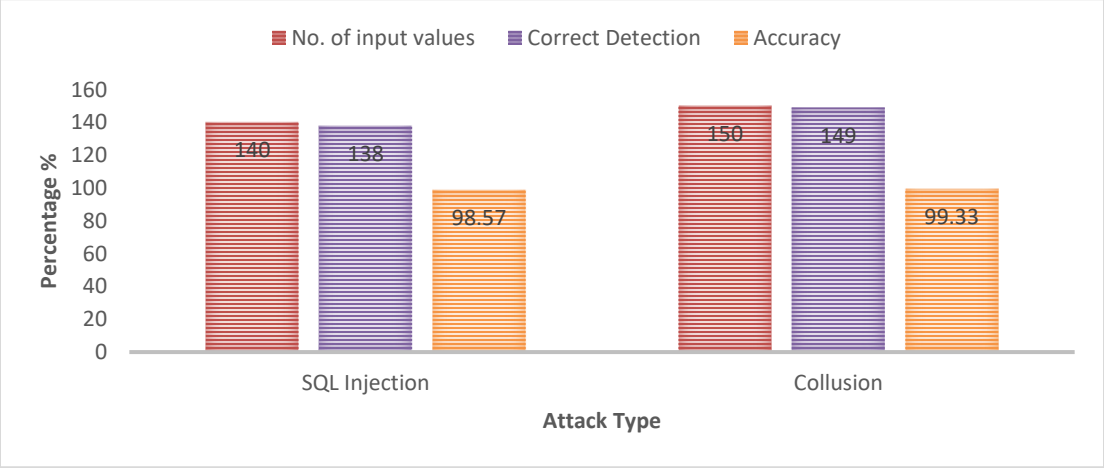

Fig 3: attack detection with various input types

The Figure 2 describes performance evaluation of proposed algorithms vs conventional attack detection techniques.

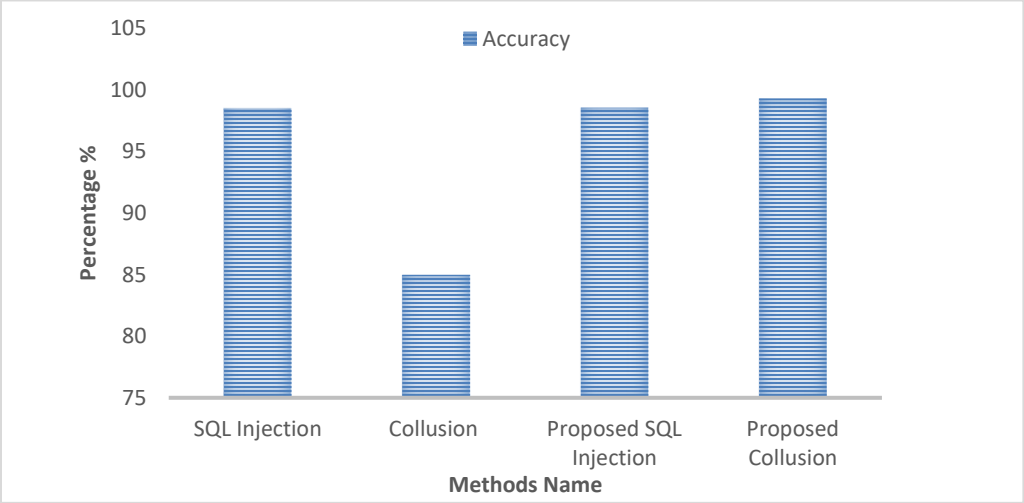

Fig 4: comparative analysis of proposed attack detection methods vs traditional methods

The SQL injection achieves similar results when evaluate with existing methods but collision attacks improvise 14% over the exiting methods.

6. Conclusions

In this paper, we present a unique method for conducting digital forensics in the cloud, using information about virtual machines (such as their IP address and mac address) as evidence. This method increases cloud performance in terms of size and time by storing information about hostile virtual machines, which are identified with the help of an intrusion detection system integrated into the virtual machine. The suggested method increases cloud performance by saving information about a questionable virtual machine in a persistent storage system. A time-saving method for finding SQL injection vulnerabilities. When used with parameters, the method retrieves the value from a SQL query attribute of web pages and checks it against the given value. This method incorporates both static and dynamic analysis. We have discussed the potential threats posed by tampered QR codes in this study. We anticipate that the hacking community will devote more time and resources to this kind of attack in the future as the usage of QR codes for marketing reasons continues to grow in popularity. With the goal of testing the proposed system in a wide range of dispersed settings using fog nodes and a wide variety of input items.

References

[1] Hlaing, Zar Chi Su Su, and Myo Khaing. "A detection and prevention technique on sql injection attacks." 2020 IEEE Conference on Computer Applications (ICCA). IEEE, 2020.

[2] Chowdhury, Shreya, et al. "A Comprehensive Survey for Detection and Prevention of SQL Injection." 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS). Vol. 1. IEEE, 2021.

[3] Ismail, Safwati, Mohammed Hazim Alkawaz, and Alvin Ebenazer Kumar. "Quick response code validation and phishing detection tool." 2021 IEEE 11th IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE). IEEE, 2021.

[4] Jemal, Ines, et al. "Sql injection attack detection and prevention techniques using machine learning." International Journal of Applied Engineering Research 15.6 (2020): 569-580.

[5] Chen, Ding, et al. "Sql injection attack detection and prevention techniques using deep learning." Journal of Physics: Conference Series. Vol. 1757. No. 1. IOP Publishing, 2021.

[6] Bhoskar, Nikita, et al. "A Survey on Secrete Communication through QR Code Steganography for Military Application." Int. J. Res. Appl. Sci. Eng. Technol 10.1 (2022): 728-731.

[7] Subairu, Sikiru, et al. "A Review of Detection Methodologies for Quick Response code Phishing Attacks." 2020 2nd International Conference on Computer and Information Sciences (ICCIS). IEEE, 2020.

[8] Wahsheh, Heider AM, and Mohammed S. Al-Zahrani. "Secure real-time computational intelligence system against malicious QR code

links." International Journal of Computers, Communications and Control 16.3 (2021).

[9] Hu, Jianwei, Wei Zhao, and Yanpeng Cui. "A survey on sql injection attacks, detection and prevention." Proceedings of the 2020 12th International Conference on Machine Learning and Computing. 2020.

[10] Arock, Michael. "Efficient Detection Of SQL Injection Attack (SQLIA) Using Pattern-based Neural Network Model." 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). IEEE, 2021

[11] Ravi, C., Yasmeen, Y., Masthan, K. ., Tulasi, R. ., Sriveni, D. ., & Shajahan, P. . (2023). A Novel Machine Learning Framework for Tracing Covid Contact Details by Using Time Series Locational data & Prediction Techniques. International Journal on Recent and Innovation Trends in Computing and Communication, 11(2s), 204–211. https://doi.org/10.17762/ijritcc.v11i2s.6046

[12] Prof. Nitin Sherje. (2017). Phase Shifters with Tunable Reflective Method Using Inductive Coupled Lines. International Journal of New Practices in Management and Engineering, 6(01), 08 - 13. Retrieved from http://ijnpme.org/index.php/IJNPME/article/view/50

[13] Kathole, A.B., Katti, J., Dhabliya, D., Deshpande, V., Rajawat, A.S., Goyal, S.B., Raboaca, M.S., Mihaltan, T.C., Verma, C., Suciu, G. Energy-Aware UAV Based on Blockchain Model Using IoE Application in 6G Network-Driven Cybertwin (2022) Energies, 15 (21), art. no. 8304.