American Journal of Computer Science and Engineering Survey

ISSN: 2349-7238

Open access Research Article

Adaptive Technique for Secure Data Transmission to Improving the Cyber Security

P Salman Raju*, **P Venkateswara Rao**

Department of Research Scholar, Adikavavi Nannaya University, India

ABSTRACT

Internet usage has grown many folds in the last two decades. For many businesses, web applications have become one of the most significant and critical interfaces. Throughout today's economic and social life, the use of web based services (such as e-commerce, online banking, and web based communications, to name a few) has become a common habit. Countless applications operate worldwide on millions of servers, and their numbers are steadily increasing. It has become focus of attackers and hackers for the attacks because of the huge growth of internet usage. It is necessary for all companies to develop and protect their applications in order to maintain their credibility and keep their products relevant for users. Web applications have brought in new classes of computer security vulnerabilities, such as SQL injection (SQLIA) and cross site scripting (XSS), which have exceeded previously prominent vulnerability classes in recent years, such as buffer overflows, in both new vulnerability reports and exploit reports. The primary purpose of this research is to study the vulnerabilities of SQL Injection and Cross Site Scripting and to propose an optimistic Security model for secure data transmission. In this work, I proposed a model with framework for asymmetric and symmetric encryption, which is far more reliable than the traditional methods of encryption.

Key Words: SQLIA; XSS; Cyber security; Dragonfly algorithm; AES; IBS

INTRODUCTION

In the last couple of decades, the practice of the Internet has developed many folds. Web applications have grown to be one of the most essential and significant interfaces for several businesses. The use of web based services including e-commerce, online banking, and web based emails, has grown to a wide spread routine in today's financial and common life. Many applications are working on millions of servers globally, and their numbers are continually increasing. Because of massive growth in internet usage, it grew a target to attackers and hackers for the attacks. It is simple to target less secure machines. It is apparent, attacking less secure applications are more comfortable to target the system. The complexity and extensibility of Web applications make these applications easy prey by utilizing software vulnerabilities and weaknesses. SQL infusion attackers are a sort of infusion attacker, in which SQL directions are infused into information plane contribution to request to impact the execution of predefined SQL directions [1].

Types of Security

This section will describe the different types of security. For example, many existing methods such as shifting, data stream investigation, entry testing, and guarded coding can distinguish and avert a subset of vulnerabilities leading to SQLIs and XSS.

Session hijacking: Web based applications often use sessions to upgrade the client amicable experience for their clients. Usage of different sorts of the session the board does this Session the executive chips away at the accompanying idea. Session IDs are recognizable proof tokens for the clients, and are utilized by the servers to keep up the session information (e.g., factors)

Received:	03-January-2022	**Manuscript No:**	IPACSES-21-10461
Editor assigned:	05-January-2022	**PreQC No:**	IPACSES-21-10461(PQ)
Reviewed:	19-January-2022	**QC No:**	IPACSES-21-10461
Revised:	24-January-2022	**Manuscript No:**	IPACSES-21-10461(R)
Published:	31-January-2022	**DOI:**	10.36846/IPAJCSES-10.1.3

Corresponding author Salman Raju P, Department of Research Scholar, Adikavavi Nannaya University, India; E-mail: rajupolavarapu.nird@gmail.com

Citation Salman Raju P, P Venkateswara Rao (2022) Adaptive Technique for Secure Data Transmission to Improving the Cyber Security. Am J Comp Sci Engg Surv. 10:3

[2] (**Figure 1**).

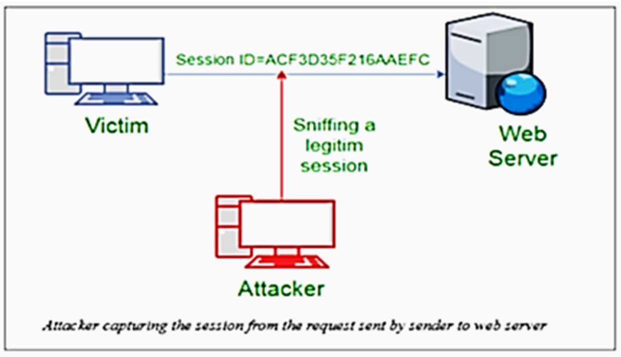

Session ID=ACF3D35F216AAEFC
Victim
Sniffing a legitim session
Web Server
Attacker
Attacker capturing the session from the request sent by sender to web server

Figure 1: Capturing user session id.

Cross Site Scripting (XSS): For instance, when a purchaser visits an online store, the articles chosen by him get included in a virtual shopping basket. He settles the request by heading off to the store's checkout page. This ordinarily includes a continuous Sniffing a real session correspondence between the customer and the server. The Session ID is one arrangement accessible to monitor the customer's truck for each expansion or erasure of articles from the store. Under specific circumstances, the Session Fixation weakness empowers the thief to execute a Session Hijacking attack, by assuming responsibility for the unfortunate casualty's session identifier esteem [3].

SQL injection: An SQL injection is a security weakness that occurs within database application layers. SQL Injection is a web application database utilisation tool. This is achieved by inserting the SQL statements as an input string to access the database unauthorized. For example, a web page on a news site may display articles related to a specific category, such as Sports, Politics, etc., depending on the value passed through the URL query string [4].

Problem statement: Web applications, for example, e-banking and electronic correspondence highlights, have turned out to be basic to the day to day lives of numerous individuals with the rise of the Internet and online utilization. Web applications have brought new kinds of vulnerabilities to PC protection, for example, SQL infusion (SQLIA) and Cross webpage scripting (XSS) that as of late have surpassed already conspicuous helplessness classes, for example, support floods in the two reports of new vulnerabilities and reports of adventures. SQL infusion and XSS are the two occasions of the more extensive class of info approval based vulnerabilities. At their center, both include one framework accepting, changing, and developing string esteems, some of which originate from untrusted and obscure sources, and showing those qualities to another framework that deciphers them as projects or program sections. These information approval based vulnerabilities in this way require on a very basic level new systems to describe and alleviate them [5].

Aim and Objectives

This research also prescribes and recommends defence methods for security attacks on web applications.

The proposed development has the following subsidiary objectives:

1. Designing characterization and runtime research of SQL injections on web applications and executing Cross Site Scripting and XPath injections. Also, this research explains about Static analysis for SQL injections.
2. This research also describes, Static Analysis for identifying Cross site Scripting (XSS vulnerabilities), empirical assessments of the XSS vulnerability attacks. This paper additionally describes implementations of secure algorithms, showing them to be effective for their respective settings [6].

Literature Review

The growth of technology equally increases vulnerability. Subsequently, the more researches center around innovation, the more importance laid on the detection and prevention of related vulnerabilities. There is a rich amount of literature in this field.

Review of existing studies: SQL injection and cross site scripting (XSS) has emerged as the recent vulnerability in the web world. In such a situation, it is important to give extra protection systems to secure the critical data that is recovered by SQL queries developed cautiously by programmers [7].

SQL injection attacks: SQL injection attacks (SQLIAs) Structured query language (SQL) is a deciphered language that is used in web applications powered by a database that generates SQL articulations that fuse the client's knowledge or content.

Cross site scripting attacks: Cross website scripting (XSS) is another web application procedure of attacking that includes reverberating attacker provided code into a customer's program by means of site pages seen by the objective clients.

Comparison of existing studies: Many research studies have been directed in addressing the issues identified with SQLI and XSS since the time they were discovered. Since it is very difficult for eliminating all the vulnerabilities of XSS and SQLI prior to being deployed in an application, penetration testing and other powerful investigation systems ought to be utilized after deploying to persistently test the application [8] (**Table 1**).

Table 1: Comparison of some of the studies discussed [12].

Author	Objective	Techniques developed	SQLI or XSS	Results obtained	Research gap
Ciampa et al. (pp. 43-49)	Identifying vulnerabilities of SQLI in web applications.	Heuristic-based approach and a tool-named V1p3R for penetration testing of the web applications.	SQLI	The developed tool reduced the cost by restricting the consumption of resources and interaction of the testers.	Improvement in tool heuristics, specifically concerning the correct interpretation of web pages that are a result of attacks.

Sharma et al. (pp. 343-351)	Prevention of SQLI and reflected XSS attacks on the PHP website.	Modification of the existing model by employing the the logic for averting the different types of SQLIA, integration of a separate module for averting reflected XSS attacks.	Both	Results showed that the developed approach was 100% efficient in averting SQLI and reflected XSS attacks.	Additionally, keeping in perspective on the adequacy of this model in averting specific kinds of Reflected XSS attacks, the model could be additionally extended to forestall different sorts of XSS attacks.
Ghafarian (pp. 833-838)	Reducing the SQLIA vulnerability in web applications	A combination of static and dynamic approach is developed.	SQLIAIn	In comparison with the existing developed technique is more effective since it had the ability to handle queries of any kind. Also, the algorithm is independent of the platform.	The proposed method is in its theoretical stage and practical implementation is yet to be carried out.
Shar et al. (pp.1767-1780)	Provide an alternative solution for the current taint analyzers by introducing attributes of static code that could be utilized in predicting the particular statements of the program, instead of software components, that might be vulnerable to XSS or SQLI.	Development of vulnerability models of prediction from the historical data whichreflects developed static attributes along with vulnerability data that is known for predicting XSS and SQLI vulnerabilities.	Both	Recall of 93% recall, a false alarm of 11% to predict the vulnerabilities of SQLI, Recall of 789% recall, a false alarm of 6% to predict the vulnerabilities of XSS.	The developed predictors were application-specific.
Gupta et al. (pp.1595-1602)	Investigate the response of HTTP for extraction of script content.	Defensive XSS framework for the cloud platforms detecting the transfer of reflected XSS worms in the applications of web employed in the virtual machines present in the cloud.	XSS	The exploratory outcomes uncover that the proposed system distinguishes the XSS worms with a low pace of false negatives and positives.	Integration of the capabilities of the developed model on the different cloud computing platforms.
Parvez et al. (pp. 186-191)	Evaluating the black-box web application scanners efficiency for detecting stored vulnerabilities of SQLI and XSS.	A custom test-bed was developed for evaluating the capability of the scanner to detect stored vulnerabilities of SQLI and XSS.	Both	The detection rate of the scanners for detecting XSS attacks had improved while that for SQLIA requires improvement.	The selection of correct vectors for detecting and exploiting the stored vulnerabilities of SQLI and XSS is a significant challenge for the black-box scanners.
Gupta et al. (pp. 199-203)	Applying IDS in detecting the XSS attacks.	Rules for XSS attack detection utilizing IDS. monitored the outgoing and incoming packets for further matching with the rules of the database.	XSS	The system proposed selected the attacks of XSS precisely.	The detection speed requires enhancement, this can be accomplished by the use of other keywords along with PCRE.
Akbar et al. (pp. 286-292)	Development of OWASP ModSecurity Core Rule Set that can aid the administrator in securing the servers of the web.	OWASP ModSecurityWeb Application Firewall.	Both	The developed method obtained a 100% detection rate for both XSS and SQLI attacks.	Extension of the set of core rules of OWASP ModSecurity for securing certain XSS attacks that Was not provided by OWASP ModSecurity.

Methodology

Collection of Database and Sql Injection and Xss

SQL injection attacks: An SQL injection is a type of attack in which the attackers insert Structured Query Language (SQL) code into the inputs of the web in order to access (or modify) data. Several tools are available to detect and prevent these vulnerabilities.

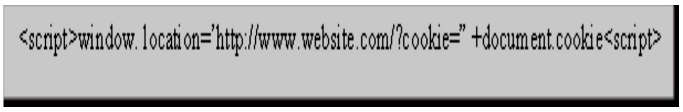
Classification of SQL injection attacks: An SQL injection is a type of attack in which the attackers insert Structured Query Language (SQL) code into the inputs of the web in order to access (or modify) data. Several tools are available to detect and prevent these vulnerabilities.

SQL injection is a model which exploits a web application's database. The information provided is description of attacks with SQL injection [9].

1. **Order wise:** It composes of first order injection attack and second order injection attack. The common data types such as DATE and NUMBER which is considered as exploitable.
2. **Classical SQL injection attack:** This attack combines with two SQL queries and then the attacking process is initiated. It does not allow accessing the additional data in a certain table.
3. **Blind SQL injection attack:** It is different from that of the original. This helps the threat agents to create the database by analysing the expressions. It is further classified as: (a) Standard Blind. It embeds operator and separator <".">
With application of <union>. (b) Double Blind SQL injection: It returns error messages from queries that removed a web application from the returned list.
4. **Attacks against database:** (a) SQL manipulation: It is being categorized as a tautology, Inference, basic union queries and piggybacked queries. (b) Code Injection: This is easily applied to the Microsoft SQL applications. (c) Functional call injection: These function calls can be used to make operating system calls or manipulate data in the database.
5. **Cross site scripting attacks:** Cross Site Scripting is one of the injection attacks that permit the hacker to execute the scripts in the user's browser. In some cases, the user becomes a victim when they visited the web pages without prior knowledge about it (or) not.

The following script is widely used for detecting XSS attacks (**Fig-**

ure 2).



```
<script>window.location='http://www.website.com/?cookie=" +document.cookie<script>
```

Identity Based Encryption (IBE): The identity attributes of the users were used for encryption and decryption purposes. The identity attributes are the email address (or) phone numbers that were deployed instead of digital certificates [10]. No prerequisites are required for the part of the recipient to receive an encrypted message [11]. No need for maintaining the public key infrastructure [12]. Inherent key escrow issue is tackled in the IBC environment [13].

i. **Set up:** It is executed by the key authority.

ii. **Private key generation:** It is executed by the key authority which takes the inputs (mpk, msk), identity id and state st.

iii. **Decryption key generation:** It is executed by the receiver.

iv. **Encryption:** This algorithm is run by the Encryption algorithm of the sender.

v. **Decryption:** It is executed by the receiver.

Advanced Encryption Standard (AES)

It is helpful for defining protection algorithms for unclassified data. AES is significantly used in the private sector [14].

The minimum requirement of AES is:

A. The algorithm must be symmetric in the key management system.

B. It must be a block cipher.

C. It should be capable of key block combination with sizes of 123-128, 192-128, 256-128 bits.

The evaluation criteria for AES are

a. **Security:** It aims to enhance the security weakness of DES.

b. **Cost:** The reasons behind these criteria are to measure and minimize the costs of licensing requirements, computational efficiency and the memory requirements.

c. **Algorithm and implementation characteristics:** It includes criteria such as flexibility, hardware, and software suitability and simplicity.

The effectiveness of the AES algorithm is computed as:

Effectiveness=f (Efficiency, Security)

Efficiency=f (performance, resource utilization)

Security=f (security margin, ancestry, and simplicity).

Proposed Frame work

Dragonfly optimization: Inspired by the swarming behaviour of the dragonfly, a novel bio-inspired optimization algorithm, named, dragonfly optimization. It belongs to the class of adaptive technique and the unconstrained test function using localization models. It holds quality features of both static and dynamic swarming process that covers vast area towards targeted destination (or) optimization [15]. Dragonfly optimization is proposed by Seyedali Mirjalili, admired by the swarming behaviour of the dragonflies. Actually, it flies in a different direction in small groups in search of food. In technical terms, optimization in search of food is known as the exploration phase. The followings are the pictorial representation of the dragonfly algorithms [16] (Figures 3-9).

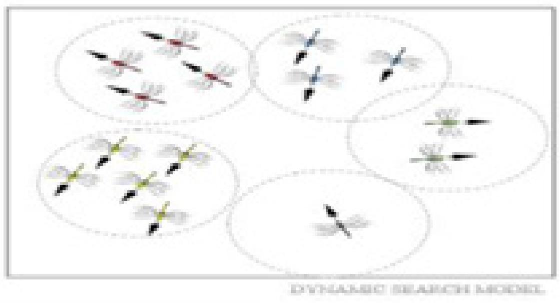


Figure 3: Dynamic search model.

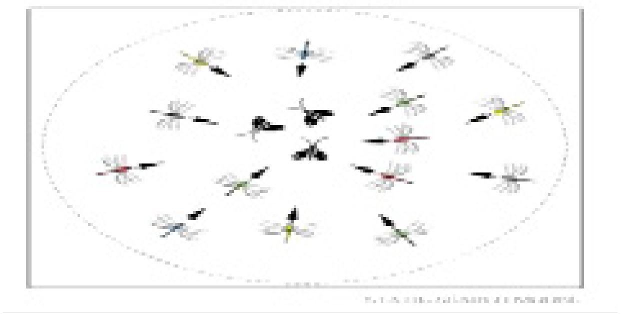


Figure 4: Static Search model.

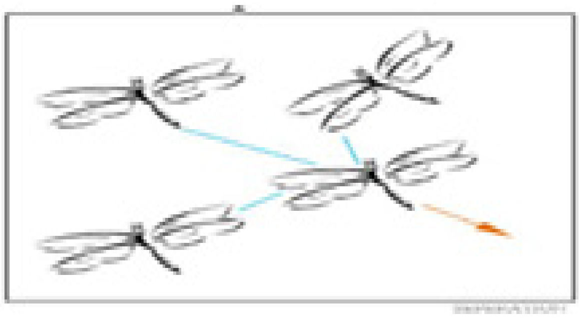


Figure 5: Separation process.

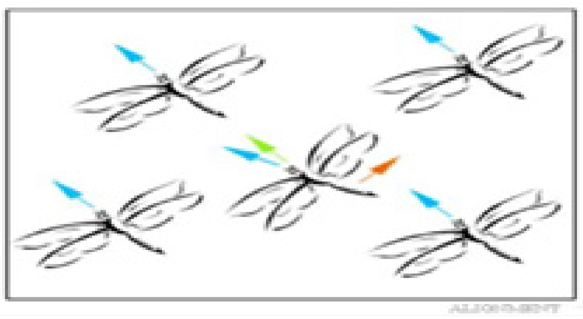


Figure 6: Alignment process.

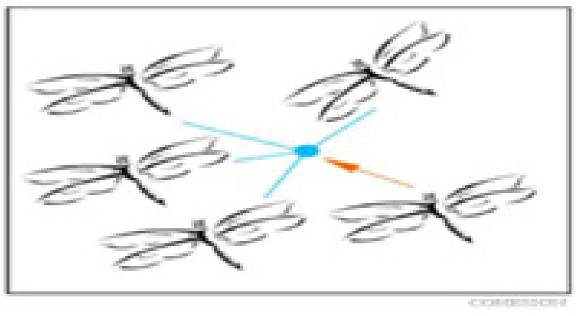


Figure 7: Cohesion process.

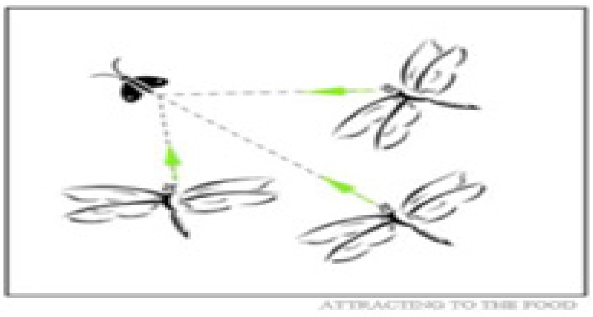


Figure 8: Attracting to the food

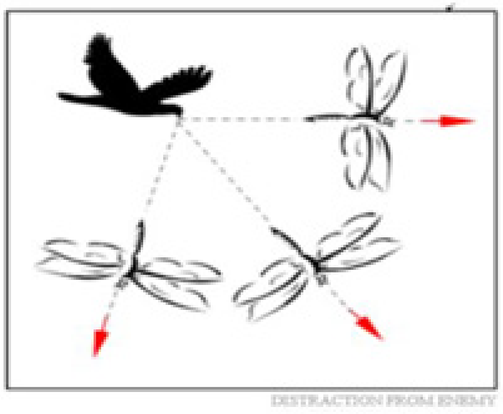


Figure 9: Distraction from enemy.

Thus, each portion of the dragonfly algorithm is formulated as:

1. The separation part is given as:

$$S_i = -\sum_{j=1}^{N} X - X_j$$

2. The alignment part is given as:

$$A_i = \frac{\sum_{j-1}^{N} X_j}{N}$$

3. The cohesion part is given as:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X$$

4. The attraction towards food source is given as:

$$F_i = X^{+} - X$$

5. Step vector is given as:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_t$$

6. The position vector is given as:

$$X_{t+1} = X_t + \Delta X_{t+1}$$

7. Finally, the position of the dragonfly is updated as follows:

$$X_{t+1} = X_t + Levy(x) * X_t$$

Where,

X represents the location of the current individuals.

N represents the neighbouring individuals

X+ represents the position of food source

X- represents the position of the enemy s represents the weight of the separation process

a represents the weight of the alignment process

c represents the weight of the cohesion process

f denotes the weight of food source

e denotes the weight of enemy

w denotes inertia weight

t represents iteration counter

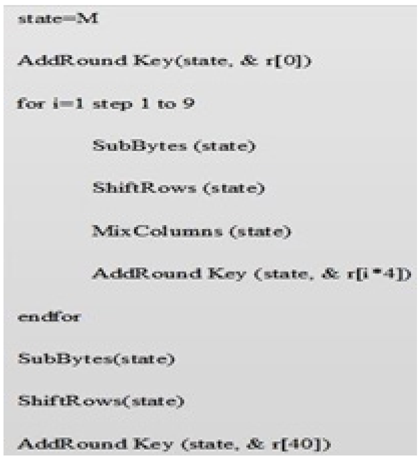
d represents the dimension of position vectors that levy calculated flight steps.

The followings are the pseudo code of the proposed algorithms to detect and prevent SQL injection attacks and cross-site scripting.

A. Pseudo code for AES (Figure 10)

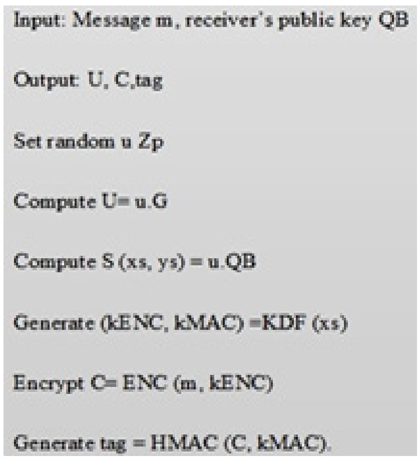
B. Pseudo code for IBE (Figure 11)

C. Pseudo code for DA-GA Algorithm (Figure 12)



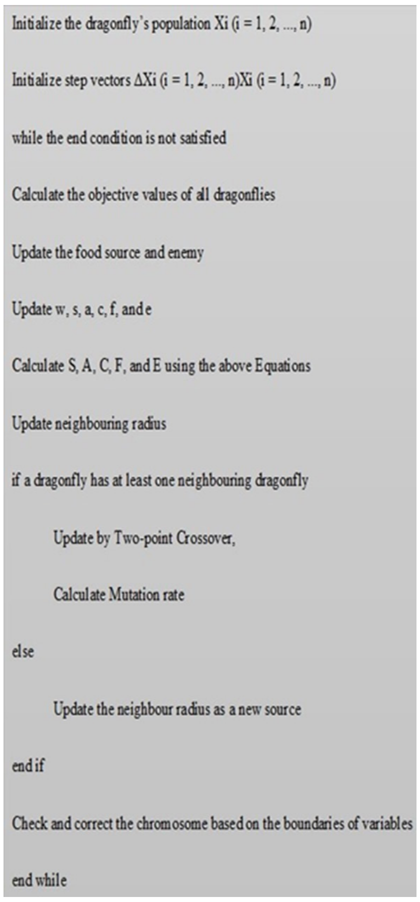
```
state=M
AddRound Key(state, & r[0])
for i=1 step 1 to 9
        SubBytes (state)
        ShiftRows (state)
        MixColumns (state)
        AddRound Key (state, & r[i*4])
endfor
SubBytes(state)
ShiftRows(state)
AddRound Key (state, & r[40])
```

Figure 10: Pseudo code for AES.



```
Input: Message m, receiver's public key QB
Output: U, C,tag
Set random u Zp
Compute U= u.G
Compute S (xs, ys) = u.QB
Generate (kENC, kMAC) =KDF (xs)
Encrypt C= ENC (m, kENC)
Generate tag = HMAC (C, kMAC).
```

Figure 11: Pseudo code for IBE.



```
Initialize the dragonfly's population Xi (i = 1, 2, ..., n)
Initialize step vectors ΔXi (i = 1, 2, ..., n)Xi (i = 1, 2, ..., n)
while the end condition is not satisfied
Calculate the objective values of all dragonflies
Update the food source and enemy
Update w, s, a, c, f, and e
Calculate S, A, C, F, and E using the above Equations
Update neighbouring radius
if a dragonfly has at least one neighbouring dragonfly
    Update by Two-point Crossover,
    Calculate Mutation rate
else
    Update the neighbour radius as a new source
end if
Check and correct the chromosome based on the boundaries of variables
end while
```

Figure 12: Pseudo code for DA-GA Algorithm.

RESULTS AND DISCUSSION

Time and Speed analysis of the algorithm determines the characteristics of the process via time required for encryption and decryption.

Encryption Time

The time required to converts the plaintext into ciphertext is said to an encryption time. The encryption time based on the message block size and key size, and represented in milliseconds. It has direct impacts on the performance of the encryption algorithm. Every cryptographic algorithm required minimum encryption time, in order to make the encryption scheme responsive and fast.

Decryption Time

The time required to recover the plaintext from cipher text is said to decryption time. For the purpose of cryptographic algorithms fast and responsive, it is desirable that the decryption timeless similar to the encryption time and it is also measured in milliseconds.

The below mentioned graph shows the effective performance of the algorithm

The comparison graph showed that the test data showed higher number of counts in comparison to the sample data as well as SQL database.

CONCLUSION

This research appears to include approval based vulnerabilities that exist with regard to meta-programming in the specific circumstances of web applications, or even more so for the most part. In light of ideas from programming dialects and compilers, this paper gives the primary principled portrayal for such vulnerabilities, with formal definitions specifically for SQL injection and XSS. In addition, increasing on this portrayal, the exploration contributes a reasonable algorithm for runtime security, static investigation, and test based investigation of web applications to identify vulnerabilities in application code and prevent misuse of attackers.

The vulnerabilities are normally present in web applications. It appears to be misused by SQL injection attacks to collect the gain from the application's involvement. Careful coding has been recommended as an answer to mitigating the SQLIAs.

Since Cross Site Scripting (CSS) and SQL Injections (SQLI) are the extremely risk filled that happens regularly in the present web attacks, we tried to focus more on them and attempted to clarify with point by point models. Further we suggested a framework to differentiate between SQLIA and XSS based on AES and IBE encryption. Additionally, it is streamlined depending 130 on the neural relapse systems. What's more, we executed the SQLIA and XSS model location program. Our pages of structure inspections that can trigger XSS and gain powerlessness follow during the position procedure. In line with these, our approach will naturally produce attack vectors, which is very important for engineers and analysers to identify as well, to overcome the vulnerabilities in the code.

REFERENCES

1. Benjamin L, Úlfar E (2007) Using web application construction frameworks to protect against code injection attacks. Proceedings of the 2007 workshop on Programming languages and analysis for security. ACM.
2. Desmet L (2008) Provable protection against web application vulnerabilities related to session data dependencies. IEEE transactions on software engineering. 34(1):50-64
3. Kunal G, Rajni RS, and Manish D (2017) Cross site scripting (XSS) attack detection using intrusion detection system. 2007 IEEE
4. Ahmad G (2017) A hybrid method for detection and prevention of SQL injection attacks.2017 Computing Conference. IEEE
5. Guardia Vieira M, Madeira H, Fonseca J (2007) Web Vulnerability Scanning Tools for SQL Injection and XSS Attacks Dependable Computing. CISUC - Polytechnic Inst. of Guardia,Testing and Comparing, 2007. PRDC 2007.13th Pacific Rim International Symposium on Date of Conference: 17-19 Dec. 2007.
6. Umasankari P, Uma E, Kannan A (2013) Dynamic Removal of Cross Site Scripting Vulnerabilities in Web Application. International Journal of Advanced Computational Engineering and Networking. Volume 1(4).

7. Subhranil S, Sapna S, Ritu K (2016) Study on sql injection attacks: Mode detection and prevention. International Journal of Engineering Applied Sciences and Technology, Indexed in Google Scholar, ISI etc., Impact Factor: 1.494 1.8 (2016): 23-29.

8. Farmer, Randall F, Chip M, Douglas C (1994) From Habitat to global cyberspace. Proceedings of COMPCON'94. IEEE, 1994

9. Jivesh G, Jivika G (2007) Ramifications of cyber-crime and suggestive preventive measures. IEEE International Conference on Electro/Information Technology.

10. Hydara I, Sultan ABM, Zulzalil H, Admodisastro N (2014) Current state of research on cross-site scripting (xss) c a systematic literature review. Information & Software Technology 58 (2014) 170–186 [CrossRef]

11. Saleh A, Mohamed A, Hemeida A, Ibrahim A (2018) Comparison of different optimization techniques for optimal allocation of multiple distribution generation. Proceedings of the 2018 International Conference on Innovative Trends in Computer Engineering (ITCE), Aswan, Egypt, February 2018

12. Parmar A, Darji P (2018) Comparative analysis of optimum capacity allocation and pricing in power market by different optimization algorithms. Advances in Intelligent Systems and Computing. pp. 311–326. Springer, Singapore

13. Halfond, William G, Jeremy V, Alessandro O (2006) A classification of SQL-injection attacks and Countermeasures .Proceedings of the IEEE International Symposium on Secure Software Engineering. Vol.IEEE

14. Apon D, Fan X, Liu FH (2016) Compact identity-based encryption from lwe. IACR Cryptology ePrint Archive.

15. Ritu G, Ravi B (2014) Protection against SQL Injection Attack on Web Applications Using AES and Stored Procedure. IJARCSSE. 4(5):2014

16. Diab AAZ, Rezk H (2018) Optimal sizing and placement of capacitors in radial distribution systems based on grey wolf, dragonfly and moth-flame optimization algorithms. Ir J Sci Tech Trans Elec Engg. 43(1).pp. 77–96.