Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

Protection Against SQL Injection Attack in Cloud Computing

B. Shunmugapriya, Dr. B. Paramasivan
National Engineering College
Nallatinputhur, Tamil Nadu

***Abstract*:- Cloud Computing is a promising paradigm that allows customers to obtain cloud resources and services according to an on-demand, self-service, and pay-by-use business model. There is the number of web application threats in cloud computing one among them is the SQL injection attack (SQLiA). In this, the attackers produce a query of their interest to have illegal access to the database. To prevent this, we use the Twofish encryption algorithm is used to secure the client's sensitive information. In this algorithm, the file uploaded by the data owner is encrypted using the Twofish algorithm and then stored it in a database.**

Keywords —Cloud computing, SQL injection attack (SQLiA), twofish encryption and decryption

I. INTRODUCTION

Cloud computing pretends a significant change in storing information and sprint application. Instead of managing programs and data on an individual desktop computer, everything is hosted in the cloud, i.e., a nebulous grouping of processors and servers accessed via the Internet. Through cloud computing we can access all our applications and documents from anywhere at any time in different locations for collaboration. Cloud is hosted by Google that consists of both small PCs and more massive servers. The cloud of Google is a private one that can be publicly accessed by Google users. Cloud can be extended beyond a single company or enterprise. Cloud serves the applications and data which are available to users, cross enterprises, and cross-platforms. Cloud can be accessed via the Internet. Any approved user can access any authorized users from any computer can access these docs and apps. But the technology and infrastructure behind the cloud are invisible to the user. There are six fundamental properties of cloud computing are User-centric, Task-centric, Powerful,Accessible, Intelligent, Programmable

A. Cloud Computing: The Next Step in Collaboration

For cloud-based projects, the users can collaborate from multiple locations within the corporation and various organizations. Google has a collection of servers that are used to power its massive search engine. On the infrastructure side, IBM, Sun Systems, and other big iron providers are offering the hardware necessary to build cloud networks. On the software side, dozens of companies are developing cloud-based applications and storage sevices.

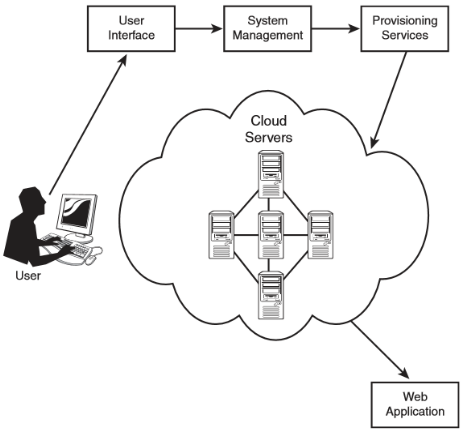


Fig.1. Architecture of cloud computing

B. Cloud Storage

Cloud computing is mainly used for data storage. Rather than using dedicated servers for storing data, multiple third-party servers in cloud storage is used. When storing data, the user observes a virtual server—that is, it appears as if the data is stored in a particular place with a correct name. But that place doesn’t exist in reality. In reality, the user’s data could be stored upon any one or more of the workstation used to generate the cloud. The cloud dynamically supervises available storage space. The advantages of Cloud storage is based on both economic and security associates. For financial, virtual resources in the cloud are typically cheaper than devoted physical resources associated with the pc or network. The data or information which is stored in the cloud is secure from accidental erasure, which is related to security. If one machine in the cloud crashes, the data is duplicated on other devices in the cloud.

C. SQL Injection Attack (SQLIA) Process

The websites like data-driven are vulnerable to SQL Injection attacks, where a database is a black box in three-tier architectures. In this architecture, the SQL statements are generated in response to HTTP requests. These HTTP requests may have parameters that are utilized by attackers to generate a query of their concerns to have illegal access to the database, as shown in Fig. 2.

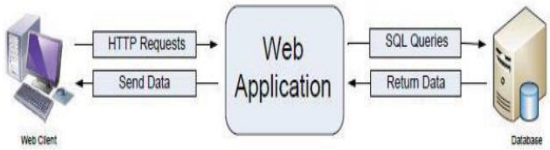


Fig. 2. SQL Injection attack process.

Fig. 4 shows the login page, which is most vulnerable for the SQL injection attack, and the following PHP code snippet produces dynamic query in response to user input, as shown in Fig. 5and 6.

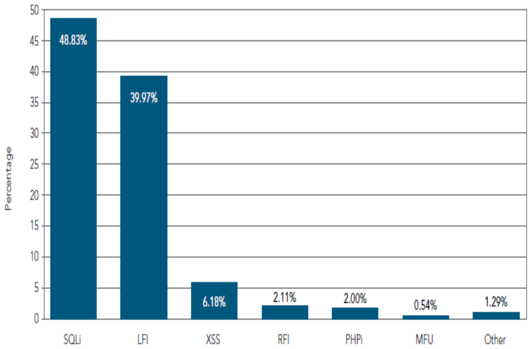
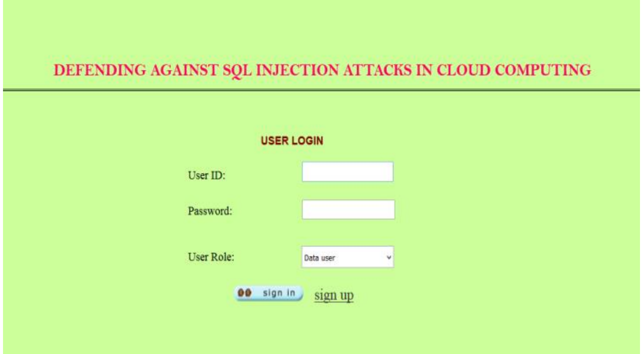


Fig.3.Web Application Attack Frequency



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

USER LOGIN

User ID:

Password:

User Role: Data user

Fig. 4.Log In form.

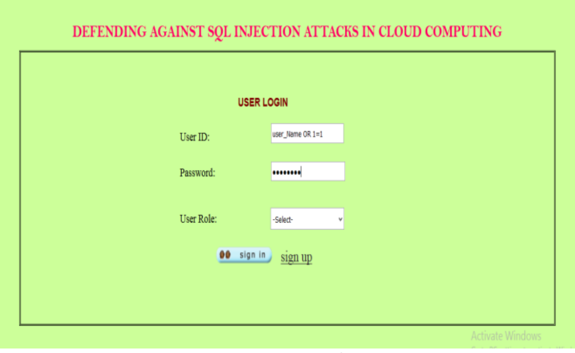
```
//connect to a database
mysql_connect(servername,username,password);
//store user input in the variables collected from the user input
login form
$username=$_POST[username];
$password=$_POST[password];
//dynamically build the query from the user input
$query=”SELECT”FROM          the_users          WHERE
username=”$username” AND password=”$password”
//execute a query
$result-mysql_query($query);
If($result)
        return true;
else
        return false;
```

Fig. 5. PHP Code snatch to generate dynamic query in response to client input.

*SELECT *FROM tbl users WHERE username=’user_Name AND PASSWORD=’pwd’;*

Fig. 6. SQL query as a result of code.

In next Fig.8 at the same form user try to attempt a simple SQLIA to bypass the authentication.



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

USER LOGIN

User ID: user_Name OR 1=1

Password: ••••••••

User Role: -Select-

sign in sign up

Activate Windows

Fig. 7.Simple attempt for SQLIA.

*SELECT *FROM tbl users WHERE usernamae=’user_Name OR 1=1 AND password=’Whatever’*

Fig. 8.Dynamic generated query in reply to above input.

In Fig. 8 attacker try to ignore the password by using the statement operator as the whole thing would be unobserved after the comments operator even the password. In these circumstances, the user name is tried to be accurate using the OR operator. This, the simple situation, and with different methods intruders want to add query of their importance to have access to them in order of their interest.

D. Techniques of SQL Injection Attack (SQLIA)

*1) **Tautology Based Attacks***

In the tautology attack, malicious contents are further using the conditional statement that, at all times, evaluates to true. The earlier scenario is the perfect example of this attack.

*Select * from tbl users Where username=’raja’ or ‘1’=’1’ and password =’anything’*

*2) **Union Attack***

In this method, the malicious query is added with a safe query using the UNION keyword .

[‘UNION SELECT pwd FROM user-info WHERE id=’abc’ and pwd=’’]

*3) **Logically Incorrect query Attack***

In this type of method, the inevitably wrong kind of query is carried out to have information about several structures of the database to proceed further.

*4) **Piggybacked Query***

Specific delimiters like “,”,”,” used to join the legal query with the illegitimate one.

*Select * from users where id=’raja’ and pwd=’’; Drop table users…’*

*5) **Alternate Encoding***

By altering the coding scheme, the illegal query can be evaded through the filter that tests the legality of the query.

*6) **Inference Attack***

Blind and timing procedures are used in assumption attacks. In blind attack, a series of uncomplicating queries are performed to have estimated about the structure of the database. In the timing attack, the query progressing time is experiential to deduce some information presented in the database.

E. Consequences of SQL Injection Attacks

To gain information about database fingerprints like the type of database, SQL language used, etc. This information helps the attacker proceeds or use more sophisticated attacks.

1) To gain knowledge about user credentials.
2) To get the database schema.
3) To extract and modify the database.
4) To carry out Denial of Services like shutting down the dropping tables, database, etc.
5) Alternate of files with false or raged information.
6) Execution of remote commands.
7) Shoplifting, account balance change.
8) Interacting with the underlying operating system.

II. TWOFISH ENCRYPTION ALGORITHM

Twofish is our submission to the AES selection process. It meets all the required NIST criteria—128- bit block; 128-, 192-, and 256-bit key; efficient on various platforms; etc.—and some strenuous design requirements,

performance as well as cryptographic, of our own. Twofish can:
• After a 12700 clock-cycle key setup, the data can be encrypted at 285 clock cycles per block on a Pentium Pro.
• After a 1250 clock-cycle key setup, the data can be encrypted at 860 clock cycles per block on a Pentium Pro.
• After a 1750 clock-cycle key setup, the data can be encrypted data at 26500 clock cycles per block on a 6805 smart card.

A.TWOFISH DESIGN GOALS

Twofish was intended to meet NIST's design criteria for AES. Specifically, they are:

- A 128-bit symmetric block cipher.
- Key length - 128 bits, 192 bits, and 256 bits.
- No feeble keys.
- Effectiveness, both on the Intel Pentium Pro and other hardware and software platforms.
- Supple design: e.g., accept additional key lengths; be executable on a wide variety of platforms and applications; and be appropriate for a stream cipher, hash function, and MAC.
- Simple design, both to make ease of analysis and ease of implementation.

Furthermore, we imposed the following routine criteria on our design has:

- Recognize any key length up to 256 bits.
- Encrypt data in less than 500 clock cycles per block on an Intel Pentium, Pentium Pro, and Pentium II, for a completely optimized version of the algorithm.
- Be able of setting up a 128-bit key (for optimal encryption speed) in less than the time required to encrypt 32 blocks on a Pentium, Pentium II and Pentium Pro.
- Encrypt data in less than 5000 clock cycles per block on a Pentium, Pentium Pro, and Pentium II with no key setup time.
- Not contain any operations that create it inefficient on other 32-bit microprocessors.
- Not comprise any activities that make it ineffective on 8-bit and 16-bit microprocessors.
- Not bring any actions that reduce its efficiency on proposed 64-bit microprocessors, e.g., Merced.
- Not include any elements that make it inefficient in hardware.
- Have a assortment of performance tradeoffs concerning the significant schedule.
- Encrypt data in less than 10 milliseconds on a product 8-bit microprocessor.
- Be executable on an 8-bit microprocessor with only 64 bytes of RAM.
- Be implementable in hardware using less than 20,000 gates.

Our cryptographic goals were as follows:

- 16-round Twofish (without whitening) should have no chosen-plaintext attack requiring fewer than 280 chosen plaintexts and less than 2 N times, where N is the key length.
- 12-round Twofish (without whitening) should have no related-key attack necessitates fewer than 264 chosen plaintexts, and less than 2N/2 time, where N is the key length.

B.TWOFISH

Twofish is a symmetric block cipher with a size of 128 bits and key size length up to 256 bits. Twofish is connected to the earlier block cipher Blowfish.

Two fish's characteristic features are the use of pre-computed key-dependent S-boxes, and a comparatively complex key schedule. One half of an n-bit key is use as the definite key of encryption and the other part of the n-bit key is used to adjust the encryption algorithm (key-dependent S-boxes). Twofish borrows some elements from other intends; Twofish has a Feistel structure like Data Encryption Standard. Twofish also utilizse a Maximum Distance divisible matrix.

Twofish is a Feistel network. It means that in each round, half of the text block is driven through an F function, and next XORed with the further part of the text block.

$$F: \{0, 1\}n/2 \text{ X } \{0, 1\}N \rightarrow \{0, 1\}n/2$$

In every round of Twofish algorithm, two 32-bit words hand out as input into the F function. Each word is wrecked up into four bytes. The four bytes are transferred through four different key_dependent substitution matrices(S-boxes). The four number of output bytes (the S-boxes contain 8-bit input and output) are joined using a Maximum Distance Separable matrix and combined into a 32-bit word. After that the two 32-bit words are united using a Pseudo-Hadamard Transform, further added to two round sub keys, then XORed with the right half of the text. There are two one-bit rotation going on, one previous to and one following the XOR. Two fish also has something named as pre-whitening and post-whitening supplementary subkeys are XORed into the content block both before the initial round and after the final round.

Each stride of the round function is bijective function. i.e., every output is achievable. We have seen too many hits or attacks against ciphers that don't have these possessions not to comprise it. The round function merges up process from different algebraic groups: S-box replacement, an MDS matrix in Galois Field GF(2^8), adding in GF(2^{32}), adding up in GF(2) (also called XOR), and 1-bit rotations. This builds the algorithm difficult to attack accurately.

The key-dependent S-boxes are designed to be challenging against the two big attacks linear cryptanalysis and differential cryptanalysis, and resistant against whatsoever unknown attacks come subsequently. Our method using this two fish algorithm which is good enough against known attacks, and enough spite to resist unfamiliar attacks. Key-dependent Substitution boxes be one way we performed that.

Key-dependent Substitution boxes were not chosen randomly, as like in Blowfish. As an alternative, the S-box construction rules, and experienced them with all possible 128-bit keys (and a subset of possible longer keys) to create sure that all the S-boxes were indeed strong. This approach permitted us to combine the strength of fixed, strong S-boxes with the strength of secret S-boxes. And Two fish has no feeble keys, as Blowfish does in reduced-round variants.

The MDS matrix was cautiously chosen to give good diffusion, to keep its MDS property even subsequent to the 1-bit rotation, and to be rapid in both software and hardware. This means that we had to investigate through all probable matrices and find the one that best gather our principles.

The PHT and key addition make available diffusion between the sub-blocks and the key. And using the Load Effective Address instruction on the Pentium processors, and that able to do all four additions in immediate two operations.

The round sub-keys are vigilantly intended, using a mechanism similar to the S-box structure rules, to avert related-key attacks and to provide good key mixing. One of the things that learned during this process is that a good key plan is not attached onto a cipher, but intended in tandem with the secret message. The 1-bit rotation is calculated to split up the byte structure; devoid of it, everything activates on bytes. This operation exists to aggravate cryptanalysts; it certainly frustrated our attempts at crypt analyzing the two fish algorithm.

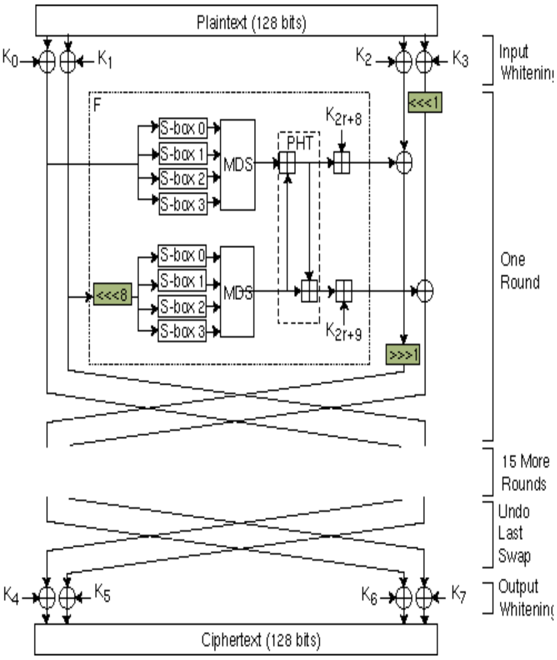


Fig.9.TWOFISH FUNCTIONALBLOCK DIAGRAM

The pre-whitening and post-whitening give the impression to add at least a round to the complexity of any attack. In view of the fact that eight XORs are smaller than a round, it creates sense to depart them in.

C. TWOFISH'S PERFORMANCE

Two fish algorithm has a selection of choices. It takes very longer for key setup and the encryption scampers earlier; this makes sense for encrypting huge amounts of plaintext with the similar key and setup the key rapidly and encryption is slower; this makes sense for encrypting a series of short blocks with rapidly changing keys.

On smart cards, two fish also have a range of transactions. The RAM calculates approximately assume that the key must be stored in RAM. If the key can be accumulated in EEPROM, then the algorithm only needs 36 bytes of RAM to run. The code size comprises both encryption and decryption code. If only encryption has to be executed, the code size and speed numbers relatively improved.

The critical key setup on this processor is about 1750 clocks per key, which can be cut significantly at the cost of two additional 512-byte ROM tables. And the 6805's be deficient in of a second index register has a significant impact on the code size and performance of the two fish algorithm; a processor with multiple index registers will be an improved fit for the algorithm.

These approximations are for a key of 128-bit. For larger keys, the added code size is insignificant: with a reduction of 100 bytes for a 192-bit keys, and less than 200 bytes for a key of 256-bit. The encryption time amplifies by less than 2600 clocks for a 192-bit key, and about 5200 clocks for a 256-bit key. Likewise, the key list pre-computation enlarges to 2550 clocks for a key of 192-bit, and to 3400 clocks for a 256-bit key.

The plaintext is dividing into four 32-bit words. In the input whitening step, these are XORed with words of four key. This is followed by 16 rounds. In every round, the two words on the left are used as input to the *g* functions. The *g* function consists of four byte-wide key-dependent substitution boxes, followed by a linear mixing step support on an MDS matrix. The solutions of the two *g* functions are combined using a PHT, and two keywords are added. These two results are then XORed into the words on the right (one of which is turn around left by 1 bit first, the other is rotated right afterwards). The left and right bisects are then swapped for the subsequent round. Subsequent to all the rounds, the exchange of the last round is reversed, and the four words are XORed with four more key words to produce the ciphertext. More properly, the 16 bytes of plaintext p_0……… p_{15} are primary split into 4 words P_0,…, P_3 of 32 bits each using the little-endian convention.

$$P_{j=} \sum_{j=0}^{3} P(4i+j)2^{8j} \quad i=0,\ldots\ldots,3$$

In the input whitening step, these terms are XOR with 4 words of the expanded key.

$$R_{0,i} = P_i \oplus K_i \qquad i = 0, \ldots.,3$$

In every 16 rounds, the first two words are used as input to the function F, which also takes the round integer as input. The third word is XOR, with the first output of F and then revolved right by single bit. The fourth word is revolved left by one bit and then XOR with the second output word of F. At last; the two halves are exchanged.

$$(F_{r,0}, F_{r,1}) = F(R_{r,0}, R_{r,1}, r)$$

$$R_r + 1{,}0 = ROR\left(R_{r,2} \oplus F_{r,,0}, 1\right)$$

$$R_r + 1{,}1 = ROL(R_{r,3}, 1) \oplus F_{r,1}$$

$$R_r + 1{,}2 = R_{r,0}$$

$$R_r + 1{,}3 = R_{r,1}$$

intended for r = 0……… 15 and where ROR and ROL are methods that rotate their 1st parameter (a 32-bit word) right or left by the bit numbers specified by their second argument. The output whitening step undoes the exchange of the last round, and XORs the data words with 4 words of the expanded key.

$$C_i = R_{16,(i+2) mod\ 4} \oplus K_{i+4} \quad i = 0, \ldots,3$$

D. CRYPTANALYSIS OF TWOFISH

Our attack works against five rounds of Two fish, without the pre-whitening and post-whitening. It requires 222.5 chosen-plaintext pairs and 251 works. We anticipate further research and techniques will extend this attack a few more rounds, but don't believe that there are any attacks against in excess of nine or 10 rounds.

We also comprise a related-key attack. It's a fractional chosen-key attack on 10 rounds of Two fish without the pre-whitening and post-whitening. To increase the attack, we have a pair of related keys. We acquire to choose 20 of the 32 bytes of every key. We have absolute control over those 20 bytes of both keys. We don't know the enduring 12 bytes of essential, but we do know that they are the same for both keys. We end up trying about 264 chosen plaintexts under each key and doing about 234 works to recover the remain unknown 12 bytes of the key. No, it's not a realistic attack, but it's the best we can do. And we have reduced-round attacks on simplified variants: Twofish with fixed S-boxes, Twofish without the 1-bit rotations, and so on.

III.SURVEY PAPER

Ensuring data storage security in cloud computing

The main objective of this paper is to resolve the security issues that are to avoid illegal access; it can be done with the assist of a distributed scheme by using homomorphism token to give security of the data in the cloud.

Drawbacks:

The servers are having got to necessary to operate on specified rows to check accuracy and verification for the computation of requested token.

Privacy - protecting public auditing for data storage security in cloud computing

This paper suggests a secure cloud storage system that is underneath privacy-preserving public auditing. The TPA is to carry out audits for multiple users simultaneously and efficiently.

Drawbacks:

It can endow with weaker security representations.

Hidden attribute-based signatures withoutanonymity revocation

This paper presents hidden attribute based signature from pairings.

Drawbacks:

It can provide some reset attacks.

Dynamic audit services for reliability verification of outsourced storages in clouds

This paper proposes an active audit service for verifying the integrity of untrusted and outsourced storage.

Drawbacks:

It can provide a small, constant amount of overhead.

Provable data possession at untrusted stores

This paper introduces a model for provable data possession (PDP) that permits a client that has stored data at an untrusted server to confirm that the server possesses the original data without retrieving it. The form generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs.

Drawbacks:

It must require a small, constant amount of communication per challenge.

IV. PROPOSED SOLUTION

In this article, a solution is proposed that is based on two fish encryption algorithm.

A. Proposed Solution Architecture

The proposed model, based on the two fish that works on the owner who uploads the data, will be encrypted from the vulnerabilities that occur from the hacker. While encrypting the key that is generated will be visible only to the user. When the data user wishes to download the file uploaded by the user, he requests the generated key to the owner. By accepting the request, the owner provides it to the user to decrypt the data that is downloaded. The file uploaded by the owner will be stored as an encrypted file. So that the attacker cannot hack the data from the database.The complete flow of proposed solution is shown in Fig 11.

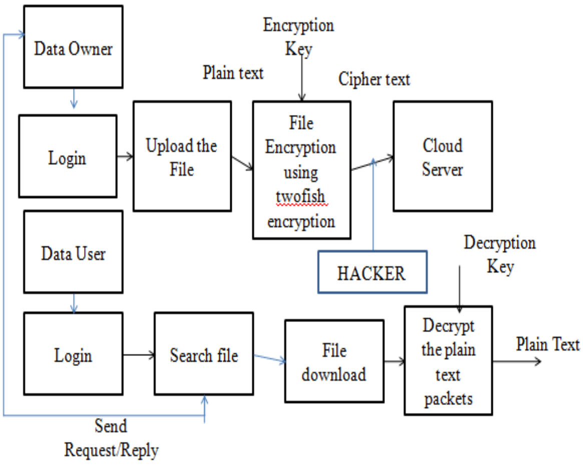


Fig.11.Data flow diagram for proposed solution.

B.ALGORITHM OF PROPOSED SOLUTION

```
Private string Encrypt(string clearText)
{
   string EncryptionKey=”MAKV2SPBNI99212”;
   byte[] clearBytes=Encoding.Unicode.GetBytes
                                    (clearText);
   using(Aesencrytor = Aes.create())
 {
   Rfc2898DeriveBytes pdb = new Rfc2898DeriveBytes
   (EncryptionKey,new  byte[]  {0x49,  0x76  ,  0x61,  0x6e,
0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64,0x65,0x76});
encryptor . Key = pdb.GetBytes(16);
   using (MemoryStreamms = new MemoryStream())
   {
     using (CryptoStreamcs = new CryptoStream (ms,
encryptor.createEncryptor(), CryptoStream.Write))
   {
cs .Write(clearBytes,0,clearBytes.Length);
cs .Close();
   }
clearText = Convert.ToBase64String(ms.ToArray());
   }
   }
  return  clearText;
   }
```

C.IMPLEMENTATION AND EVALUATION OF PROPOSED SOLUTION

To estimate the proposed solution, its performance is compared with the previous sections. These algorithms and proposed solutions are applied to find the SQLIA and block the SQLIA in various types of web application specified in Table 1.

1) IMPLEMENTATION

Using ASP.Net different classes of web Applications are used to evaluate the various tools against the separate SQL Injection Attacks.

Using ASP.Net different classes of web Applications are used to evaluate the various tools against the separate SQL Injection Attacks.

2) EVALUATION SCENARIOS

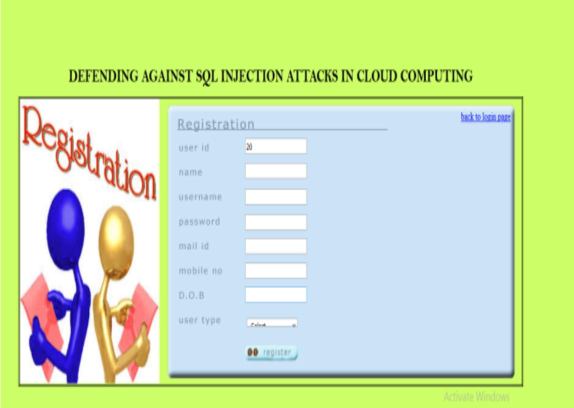
The following criteria are used to judge the performance of the Twofish encryption algorithm.

1) Registration of data owner or data user
2) Uploading the file to encrypt
3) Encryption of data in the file
4) Downloading the file
5) The decryption of the file
6) Data stored in the database

The following dataset is used to evaluate the conditions as mentioned above.

Applications	No Of Inputs
Portal	100
Online Shopping	100
University Database	100
Financial Database	100

D. Evaluation Results



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

Registration

back to login page

user id: 20

name:

username:

password:

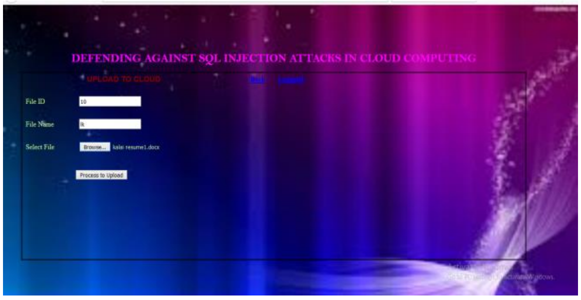
mail id:

mobile no:

D.O.B:

user type: Select

Fig.12. Registration process for proposed solution



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

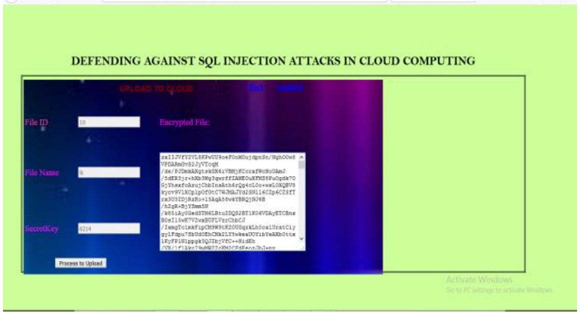
UPLOAD TO CLOUD

File ID: 10

File Name: lk

Select File: Browse... kalai resume1.docx

Fig.13. Uploading the file for encryption



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

UPLOAD TO CLOUD

File ID: 18

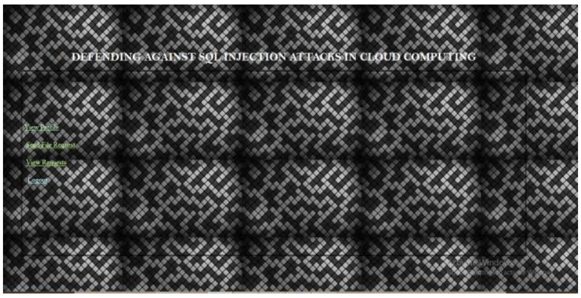
File Name: lk

SecretKey: 6214

Encrypted File:

```
zxIIJVfYZVL8KPwUU9oeFOoMOujdpnSn/HghOOw8
VPDARmGv82JyVToqM
/de/PJDmkAXgtzkGX6iVBMjKCcrxfWcNoUAmJ
/5dER3jr+hXb3Wg3qwrffZAMEOuXFMS5PuOgdk7G
GjYhsxfoAzujCbbInsAth6rQg4cL0o+wsL0XQBV8
kyov9V1XCplp0f0tC7WJMAJYd2SN1l6CZp6C2SfT
zx3U3ZDjRzKo+15AqA5EwkYBBQjSJ6B
/hZgR+BjYSmm5N
/k85iAy0GedSTM6LBtuZDQS2BT1K04VDAyETCBnx
B0sI15wH7V2wxBUFLVzrCbbCJ
/IamgTolmkFipCM9W9tKZOUGqzkLh0sa1UratCiy
gy1Fdpu7SbUd0EhCMk2LY5wkeaUOYibYeAXbOttx
lKyFP1N1ppgk3QJZbjVfC++HidEh
/VX/1flAkc79wMN22dMM2CEdFeozJhJ+nv
```

Fig.14. Encryption of file using twofish algorithm



DEFENDING AGAINST SQL INJECTION ATTACKS IN CLOUD COMPUTING

Fig.15. Data user for downloading the file

Fig.16. Server in cloud computing

V. CONCLUSION

Among many other web application threats, SQL Injection Attack has emerged as significant threats. Many solutions were proposed to detect SQLIA vulnerabilities in web applications. The proposed solution based on the Twofish algorithm has performed well to detect and block the SQLIA. One significant advantage of the proposed solution is that it can handle the advanced SQLIA techniques as the knowledge base is updated to handle modern types of threats.

VI. REFERENCES

[1] Ciampa, C. A. Visaggio and M. D. Penta, "A heuristic-based approach for detecting sql-injection vulnerabilities in web applications," in In Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems, SESS "10, New York, NY, USA,2010.

[2] A. Tajpourand ..Shooshtari, "Evaluation of sql injection detection and prevention techniques," in Computational Intelligence, Communication Systems and Networks (CICSyN), 2010 Second International Conference,2010.

[3] A. Ciampa, C. A. Visaggio and M. D. Penta, "A heuristic-based approach for detecting SQL-injection vulnerabilities in web applications," in Proceeding SESS '10 Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems, New York,2010.

[4] B. Indrani and E. Ramaraj., "X–log authentication technique to prevent sql injection attacks," International Journal of Information Technology and Knowledge Management ., vol. 4, pp. 4:323–328,,2011.

[5] A. Tajpour, S. Ibrahim and M. Masrom, "SQL injection Prevnetion and detection Techniques," International Journal of Advancements in Computing Technology, vol. 3, no. 7, pp. 85-91, August2011.

[6] D. Das, U. Sharma and D. Bhattacharyya, "An approach to detection of sql injection attack based on dynamic query matching," International Journal ofComputer

[7] A. Moosa, "Artificial Neural Network based Web Application Firewall for SQL Injection," World Academy of Science, Engineering and Technology, vol. 40, pp. 42-51, April 2010.

[8] Jin Li, Kangio Kim, "Hidden attributes- based Signatures without anonymity revocation" IEEE transactions on cloud computing vol 4, pp.no 490-499, 2011.

[9] C.Wang, Q.Wang, K. Ren and W. Lou, "Privacy- preserving public auditing for data storage security in cloud computing" in Proc. IEEE Conf. Comput. Commun., 2010, pp.525-533.

[10] C.Wang, Q.Wang and K. Ren "Ensuring data storage security in cloud computing" International Journal of Research in Advent Technology, Vol.2, No.2, Feburary 2014 E- ISSN:2321-9637.

[11] 12. Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in Proc. ACM Symp. Appl. Comput., 2011, pp.1550–1557.

[12] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Comput. Commun. Security, 2012, pp. 598–609.

[13] Aparna. K, Jyothy Solomon, Harini . M, Indhumathi . V, "A Study of Twofish Algorithm", 2016 pp.2321-9939.

[14] Solomon Ogbomon, UwagboleWilliam, J. Buchanan, Lu Fan, "An Applied Pattern- Driven Corpus to Predictive Analytics in Mitigating SQL Injection Attack",2017.

[15] Xiang Fu and K. Qian, "SAFELI – SQL Injection Scanner Using Symbolic Execution," in Workshop on Testing, Analysis and Verification of Web Software, July 21,2008.

[16] https://www.schneier.com/academic/archives/1998/12/the_twofish_encrypti.html