RESEARCH ARTICLE OPEN ACCESS

Security Implementation on EAV model using Negative database and Shuffling.

Pooja Vartak*, Prof. Amarja Adgaonkar**
**(Department of Computer Engineering, L R Tiwari College of Engineering, Mumbai University, India*
*** (Department of Computer Engineering, K C College of Engineering and Technology, Thane Mumbai University, India*
Corresponding Author: Pooja Vartak

ABSTRACT
This Paper presents an improvised security mechanism for EAV (Entity Attribute Value) data model. EAV data model for data storage has been used in various information systems now days as it gives an advantage of data flexibility and addition and modification of new data without changing the physical database schema. In EAV (Entity Attribute Value) model uses only three columns to store the data where first column stores entity, second column stores attribute and third column stores the value of the attribute for particular entity and single generic table is used to the data. As data security and database flexibility is important in this paper we are proposing security mechanism for existing databases using concept of negative database and shuffling.

Keywords: EAV Model, Negative Database, Information Security, Shuffling.

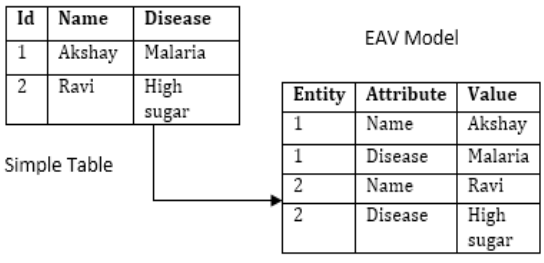
Date of Submission: 11-08-2017 Date of acceptance: 31-08-2017

I. INTRODUCTION

The Database is one of the most important assets of organizations in today's online world. According to the Verizon data breach investigation report 2015 databases are the most compromised assets that an organization has. Every day hacker releases new attacks which are designed to get access to confidential data. The primary targets of those attacks are database servers to get an access of customer's record and other confidential business data. When these hackers get access to the sensitive data they can easily obtain value, cause damage and impact on business operations. Organizations are there who need to keep their sensitive data secured from such unauthorized access, where hackers can misuse their sensitive data. Many techniques for securing database have been already implemented like encryption, hashing algorithm, etc. propose system aims to enhance the security of generic database (EAV model) by using the negative database as a main concept along with some shuffling algorithm. [1]

1.1EAV MODEL

It is required that whenever logical changes are there in database these changes should be done flexibly without changing the physical structure of database or physical schema of database. For this Entity Attribute Value (i.e. EAV) model of database can be used. The EAV pattern has a simpler approach to database design. It holds everything together in single table, rather than having a different table for each entity and its attributes. EAV model stores only non-empty value and data representation is a pair of attribute and value which describes particular entity. EAV (Entity Attribute Value) model uses only three columns to store the data where first column stores entity, second column stores attribute and third column stores the value of the attribute for particular entity and single generic table is used for the data [2].



Simple Table to EAV Model Conversion

1.2Negative Database

A negative database is a kind of database that contains huge amount of data consisting of simulating data. When anyone tries to get access to such databases both the actual and the negative data sets will be retrieved even if they steal the entire database. For example, instead of storing just the personal details you store personal details that

members don't have. Negative Database is concept of adding false data to the original data showing this data along with false data so that if hackers hacks such data even then hacker won't be able to identify the actual data..In the negative database compliment of the data is used to represent a data field, i.e. it is actually a universal data set minus the positive data information [4]. Security using the concept of negative database will add one more layer of security along with encryption/decryption and addition of erroneous data.

Shuffling

To add one more security layer along with encryption the concept of negative database is used previously [1] the algorithm used to store the false value along with original value stores the original value at fixed position suppose at 'K^{th}' position. If we want that position variable then we need to store that value along with the data as another column in the database .Which is similar to the storing of encryption key along with encrypted value which will decrease the security. Another way is to append that value with data itself which will increase algorithms complexity. But if database get hacked then hacker can easily get the value 'k' from every database field and after that by applying decryption hacker can get unauthorized access to the original data. So here by using shuffling algorithm data will be shuffled to change the positions so that no data will be there at their fixed position. This will increase security of data.

II. LITERATURE SURVEY

The proposed system tries to propose a new technique where extra layer to the security is provided by using concepts of negative database and shuffling of data to keep data safe from unauthorized users .Proposed system hide the data inside the false data so it will be difficult for hacker to extract the original data from false data after hacking of database. The proposed system will make it difficult for unauthorized users to reach to sensitive data the users.

2.1 Enhancing Privacy through Negative Representations of Data [5]

The paper introduces the concept of a negative database, in which a set of records DB is represented by its complement set. As data should be available when we needed to authorized users and should not be available to malicious parties. This paper introduce an approach to representing data that addresses some of these issues. The negative image of a set of data records is represented rather than the records themselves. Initial, assumptions are U is finite-length records (or strings), all of the same length l, and defined over a binary alphabet. The space of possible strings are logically divided into two disjoint sets: DB representing the set positive records (holding the information of interest), and U −DB denoting the set of all strings not in DB. DB is uncompressed (each record is represented explicitly), but U – DB is allowed to be stored in a compressed form called NDB.DB refers to the positive database and NDB refers to the negative database. One additional symbol to our binary alphabet is introduced, known as a "don't care," written as _. The entries in NDB will thus be l-length strings over the alphabet {0, 1, *}. The don't-care symbol has the usual interpretation and will match either a one or a zero at the bit position where the _ appears. Positions in a string that are set either to one or zero are referred to as "defined positions." With this new symbol we can potentially represent large subsets of U −DB with just a few entries.

2.2Real-Valued Negative Databases [6]

An idea for real-valued negative database is proposed in this paper, and reversing the real-valued negative database is proved to be an *NP*-hard problem. The paper proposes an algorithm for the applications where data are represented in real valued space. The algorithm uses the idea of the binary negative database as intermediate to get real valued negative database. There are three basic steps in algorithm pre-processing, encoding and decoding. In pre-processing the real valued data is converted into zeros and ones .In encoding binary negative database generation is done [3], and in decoding the reverse procedure will follow.

(1) Pre-processing: Divide the domains of attributes in DB, and convert DB to a real-valued database DBI which consists of intervals.

(2) Encoding: Encode DBI to a binary positive database DB2.

(3) Generating: Input DB2 to an algorithm for generating a binary negative database from the binary positive database such as the q-hidden algorithm or the prefix algorithm, and output a binary negative database NDB2.

(4) Decoding: Decode NDB2 to a real-valued negative database RvNDB which consists of intervals.

Since the generation algorithm for the real-valued negative database is based on the generation algorithms for the binary negative database, some more efficient generation algorithms which are dedicated to the real-valued negative database are needed.

2.4Database Security and Encryption: A Survey Study [7]

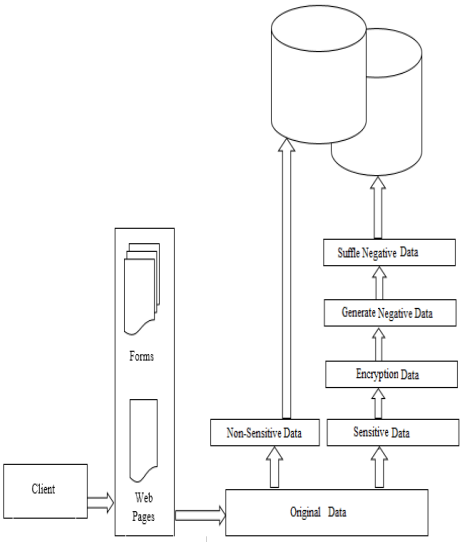
In this paper, we survey the security of database. This survey was conducted to identify the issues and threats in database security, requirements

of database security, and how encryption is used at different levels to provide the security. Information or data is a valuable asset in any organization. Almost all organization whether social, governmental, educational etc., have now automated their information systems and other operational functions. They have maintained the databases that contain the crucial information. Protecting the confidential/sensitive data stored in a repository is actually the database security. It deals with making database secure from any form of illegal access or threat at any level. Database security demands permitting or prohibiting user actions on the database and the objects inside it. Data protection and confidentiality are the security concerns. Figure below shows the properties of database security that are: confidentiality, integrity and availability

III. METHODOLOGY

3.1 Negative Database Model

While storing data to the database first sensitive and non-sensitive data will be separated and non-sensitive data will be stored directly to the database as it do not require more protection and on other hand sensitive data will passed through the further levels for providing security. At very first level encryption of the data will be performed and cipher text will passed for creation negative data and shuffling module. At this level the original encrypted data will be hided inside the false data to create the negative database .And finally negative data will be shuffled and stored in the database. While retrieving data from database the reverse procedure will be followed.[14]



In this section, the brief explanation of the working principle of the various sections which will be used in the system is mentioned.

3.2Separation of Sensitive and Non Sensitive Data:

At this stage the data will be partitioned in sensitive data and non-sensitive data. As sensitive data required security will be passed through the further steps like encryption and negative database generation to make it difficult to understand by unauthorized users and non-sensitive (does not require security) data will be stored in the database directly in the database.

Entity	Attribute	Value1	Value2	Value3

Table 4.2.1: EAV model into multiple value fields to store sensitive information

3.3Encrypt Data:

In proposed model, we first encrypt the sensitive data to be stored in the table. The actual data first passes through RSA public key encryption algorithm [c] to generate a cipher text for a given value of sensitive data. For encryption of data the input is converted to its corresponding integer value by using its ASCII value for every character of a given text. Then the text will get encrypted and pass it through RSA_Public_Key ().After that the encrypted text will be converted to base 32 to reduce its length.[1]

3.4 Negative database Creation:

The encrypted data will sent to negative database creation where we are going break the original data into n no of parts and then we are going generate some random data i.e. false data to hide the original data inside that false data to create negative data.Fig.5, describes the algorithm to convert the encrypted value of sensitive data into negative data, to provide an extra layer of security to the sensitive data.[1]

3.5 Shuffling

The proposed system uses the Fisher–Yates shuffle is an algorithm for generating a random permutation of a finite sequence—in plain terms, the algorithm shuffles the sequence. Fisher Yates shuffle algorithm is most commonly used algorithm for shuffling .since it provides unique randomness for every shuffle. It is quite efficient; indeed, its asymptotic time and space complexity are optimal. Combined with a high quality unbiased random number source, it is also guaranteed to produce unbiased results. [8].

IV. RESULTS

The security system is implemented using concept of negative database so that if someone unauthorized is able to get access of the data then also that person will not able to grab the actual data as data is mixed with huge amount of negative data. While storing data will first separated as non-

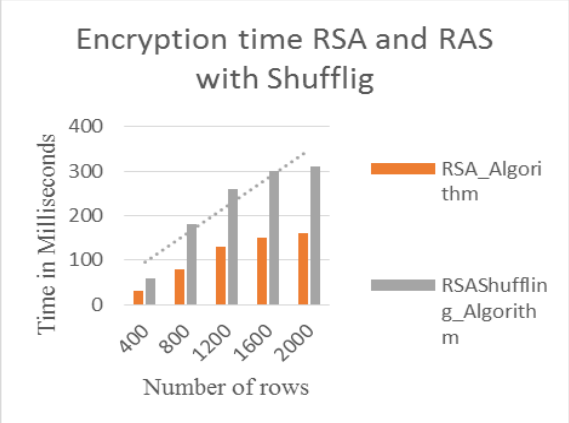
sensitive data and sensitive data will be further sent through further phases for generating negative database and shuffling.

4.1Analysis Parameters:

Plain text size Vs. Cipher text size [9]: In any cryptographic algorithm, it is essential to understand the size of the input and the size of output as this is one of the important property of an avalanche effect. Larger the size of the Cipher-text compared with the Plaintext, more secure is the Cipher-text against any Brute-Force attack.

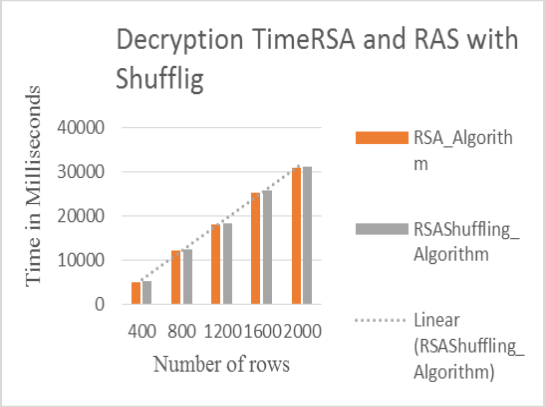
Encryption Time [9]: The encryption time is the time that an encryption algorithm takes to produce a cipher_text from a plain-text of password [9]. The encryption time has its paramount importance for varied plaintext sizes as this determines the time involved in converting plaintext into cipher-text [9]. Here as no of rows to the table the increases the size of plain text will also increase and Encryption time will also get increased for both RSA algorithm and RSA shuffling algorithm. The different set of number of rows and corresponding decryption time are tabulate below:

Number of Rows	Encryption Time in Milliseconds	
	RSA_Algorithm	RSA Shuffling_Algorithm
400	30	58
800	80	180
1200	130	260
1600	150	300
2000	160	310



Decryption time [9]: The decryption time is the time that a decryption algorithm takes to reproduce plaintext from a cipher-text [9]. The decryption time has its paramount importance for varied cipher-text sizes as this determines the time involved in converting cipher-text back to plaintext [9]. The different set of number of rows and corresponding decryption time are tabulate below:

Number of Rows	Decryption Time Milliseconds	
	RSA_Algorithm	RSAShuffling_ Algorithm
400	5100	5200
800	12200	12400
1200	18100	18400
1600	25400	25800
2000	31000	31300



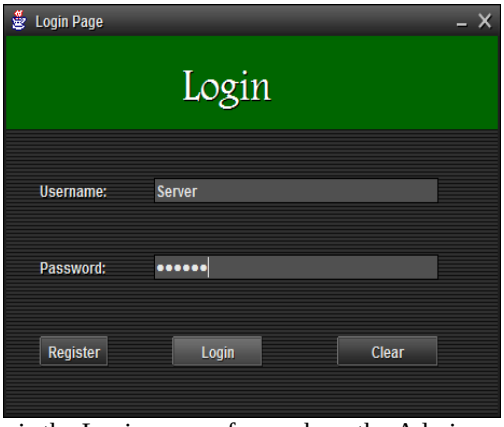
From, the experimental analysis it is observed that, Encryption as well as decryption time of RSA shuffling algorithm is quite larger than RSA algorithm. There are following reasons which justifies the same: The plain text undergoes the negative database generation block, where extra characters are added. The total number of characters to be added, is fixed. In the shuffling process the characters will shuffled to change their fixed position, to get the random positions.

Throughput [9]: The encryption time can be used to calculate the Encryption Throughput of the algorithm. The decryption computation time is the time taken by the algorithms to produce the plain text from the cipher text. The decryption time can be used to calculate the Decryption Throughput of the algorithms. Encryption time is used to calculate the throughput of an encryption scheme. It indicates the speed of encryption.

Encryption Throughput

Number of Rows	Size of Plain Text(in Kb)	Decryption Throughput	
		RSA_ Algorithm	RSA_Shuffling _Algorithm
400	8,194	1.60667	1.57577
800	16,388	1.34328	1.32161
1200	24,994	1.38088	1.35837
1600	33,600	1.32283	1.30233
2000	42,206	1.36148	1.34843

Decryption Throughput

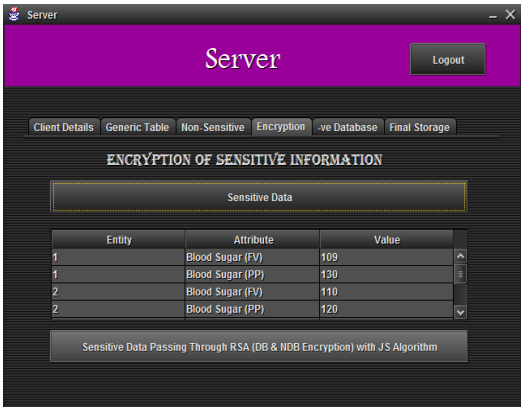


This is the Login screen from where the Admin Logins himself to the system.

Number of Rows	Size of Plain Text(in Kb)	Encryption Throughput	
		RSA_ Algorithm	RSAShuffling_ Algorithm
400	8.2	273.33	141.38
800	16.4	205.00	91.11
1200	32.8	252.31	126.15
1600	65.6	437.33	218.67
2000	131.2	820.00	423.23

V. CONCLUSION

Security has always been a domain of active research. The paper uses concept of negative database and shuffling to add more security to user's data and keep sensitive data more secure than previous methods of just using an encryption where hackers can get data by some decryption algorithms. On other hand non-sensitive data is stored in the database as it is. Which reduces the overhead of encrypting non-sensitive data .Only sensitive data will be stored as negative data which decreases the time and space required to convert non sensitive data in negative data and shuffle it.Paper explains example of medical data but concept can be applied in other domain also.



This is screen where sensitive data will be passed through encryption of data .For encryption RSA Asymmetric Encryption Algorithm is used.

REFERENCES

[1] G. Dubey, V. Khurana, S. Sachdeva"Implementing security technique on generic database". Contemporary Computing (IC3), 2015 Eighth International Conference on, Pages: 370 - 376,Year: 2015

[2] Daniel Lekberg and Patrik Danielsson "Designing and Implementing Generic database Systems Based on the entity attribute-value Model" thesis Karlstads university.

[3] Pulkit Mehndiratta, Shelly Sachdeva, Sudhanshu Kulshrestha. "A Model of Privacy and Security for EHR".DNIS 2014, Pp. 202-213.

[4] Esponda, F. (2008). "Everything that is not important: Negative databases", Computational Intelligence Magazine, IEEE, 3(2),pp. 60–63.

[5] Fernando Esponda, Stephanie Forrest and Paul Helman. "Enhancing privacy through Negative representation of data." IEEE, 2002conference.

[6] Dongdong Zhao, and Wenjian Luo "Real-Valued Negative Databases", Artificial Immune Systems – ICARIS, 2013

[7] Iqra Basharat, Farooque Azam, Abdul Wahab Muzaffar. "Database Security and Encryption: A Survey Study." .International Journal of Computer Applications. Vol. 47(12), June 2012pp. 975– 888

[8] C. Aishwarya, J. R. Beny" Novel Architecture for Data – Shuffling Using Fisher Yates Shuffle Algorithm" IJSRSET, ISSN: 2394-4099, Volume 1, Issue-6 87-390, and November-December 2015

[9] A Performance Comparison of Data Encryption Algorithms," IEEE Information and Communication Technologies, 2005. ICICT 2005. First International Conference, 2006-02-27, PP. 84- 89

International Journal of Engineering Research and Applications (IJERA) is **UGC approved** Journal with Sl. No. 4525, Journal no. 47088. Indexed in Cross Ref, Index Copernicus (ICV 80.82), NASA, Ads, Researcher Id Thomson Reuters, DOAJ.

Pooja Vartak. "Security Implementation on EAV model using Negative database and Shuffling." International Journal of Engineering Research and Applications (IJERA), vol. 7, no. 8, 2017, pp. 56–60.