Security Analysis of Reverse Encryption Algorithm for Databases

Ayman Mousa
Department of Computer Science, Workers University, Egypt.

Osama S. Faragallah and S. EL-Rabaie
Department of Computer science and Engineering, Faculty of Electronic Engineering, Menoufia University, Egypt.

E. M. Nigm
Department of Mathematics, Faculty of Science, Zagazig University, Egypt.

ABSTRACT
Encryption provides strong security for databases. To develop a database encryption strategy, many factors must be taken into consideration. Organizations must balance between the requirement for security and the desire for excellent performance. In this paper a novel encryption algorithm is proposed "Reverse Encryption Algorithm (REA)". The proposed algorithm REA is simple and yet leads to a cipher. It has achieved security and is fast enough for most applications. REA algorithm is limiting the added time cost for encryption and decryption to not degrade the performance of a database system. Moreover, designing REA algorithm has enhanced security in data encryption. Besides, the secure and performance of the proposed encryption algorithm REA is evaluated and compare with the most common encryption algorithms. Experimental results show that the proposed encryption algorithm REA outperforms other encryption algorithms at performance and security in databases. Overall, the proposed encryption algorithm REA achieves balance between the security and the efficiency.

General Terms
Database, security, algorithms, encryption.

Keywords
Database security, cryptographic algorithms and database encryption.

1. INTRODUCTION
Database encryption is a well established technology for protecting sensitive data. Unfortunately, the integration of existing encryption techniques with database systems causes undesirable performance degradation. It is a crucial technique in the security mechanisms of database. Database encryption solution is a specialized and complex and if internal resources don't have the cryptography expertise with regard to database environment, outside expertise should be used to ensure superior performance and strong security [2].

A novel innovative encryption algorithm REA is proposed. It is efficient and secure. It has accomplished security and it is fast enough for most widely used software. The proposed algorithm REA limits the added time cost for encryption and decryption and at the same time does not degrade the performance of a database system. So, analyze security and performance factors that are used as the secure and efficient criteria. Such as the key space, the key sensitivity, the security of data against attacks, the computational speed, the information entropy, and the correlation coefficient.

This paper observes a method for evaluating the security and efficiency of the proposed encryption algorithm REA and compares with the most common encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish. A comparison will be presented for those encryption algorithms for encryption and decryption times. Also, another comparison will be presented for the secure value (it is used to measure the information entropy). Furthermore, another measure of security is the correlation coefficient of encrypted fields with the proposed encryption algorithm REA.

The results of the experimental shows that the encryption and decryption time of the proposed encryption algorithm REA has a very good performance compared to other encryption algorithms. The security measure (information entropy) results indicate that the proposed encryption algorithm REA and AES are more secure than DES, 3DES, RC2, and Blowfish. Also, the correlation coefficient measure results depict that the proposed encryption algorithm REA provides more security.

The remainder of this paper is organized as follows. Section 2 discusses related work about the performance and security of the encryption algorithms. Section 3 describes the proposed encryption algorithm (REA). The performance and security factors are analyzed in section 4. Section 5 shows the experimental results for the secure and efficient evaluation of the proposed encryption algorithm REA and compares it with the most common encryption algorithms. Finally section 6 presents conclusion and future work.

2. RELATED WORK
To give more prospective about the performance of the compared algorithms, this section discusses the results obtained from other resources. It was concluded in [7] that AES is faster and more efficient than other encryption algorithms. When the transmission of data is considered there is an insignificant difference in the performance of different symmetric key schemes (most of the resources are consumed for data transmission rather than computation). Even under the scenario of data transfer it would be advisable to use AES scheme in case the encrypted data is stored at the other end and decrypted multiple times.

In [9], a study of security measure level is proposed for a web programming language to analyze four Web browsers. This study considers measuring the performances of encryption process at the programming language's script with the Web browsers. This is followed by conducting tests simulation in order to obtain the best encryption algorithm versus Web browser. In [14], Crypto++ Library is a free C++ class library

of cryptographic schemes. It evaluates the most commonly used cryptographic algorithms. Also it is shown that Blowfish and AES have the best performance among others. And both of them are known to have better encryption (i.e. stronger against data attacks) than the other two.

A study in [15] is conducted for different popular secret key algorithms such as DES, 3DES, AES, and Blowfish. They were implemented, and their performance was compared by encrypting input files of varying contents and sizes. The algorithms were tested on two different hardware platforms, to compare their performance. They had conducted it on two different machines: PII 266 MHz and P4 2.4 GHz. The results showed that Blowfish had a very good performance compared to other algorithms. Also it showed that AES had a better performance than 3DES and DES. It also shows that 3DES has almost 1/3 throughput of DES, or in other words it needs 3 times than DES to process the same amount of data.

3. PROPOSED ALGORITHM REA

A novel encryption algorithm REA is recommended, because of its simplicity, efficiency, and security. It can outperform competing algorithms. In this section provides a comprehensive yet concise algorithm. Also, gives a general analysis of the functioning of these structures.

The proposed algorithm REA is a symmetric stream cipher that can be effectively used for encryption and safeguarding of data. It takes a variable-length key, making it ideal for securing data. The REA algorithm encipherment and decipherment consists of the same operations, only the two operations are different: (1) added the keys to the text in the encipherment and removed the keys from the text in the decipherment. (2) Executed divide operation on the text by 4 in the encipherment and executed multiple operation on the text by 4 in the decipherment. Divide operation by 4 on the text to narrow the range domain of the ASCII code table at converting the text. The details and working of the proposed algorithm REA are given below.

Encryption Algorithm of the REA

The steps of the encryption algorithm REA (Figure 1) are presented in the following steps (Figure 2):

Step1: Input the text and the key.
Step2: Add the key to the text.
Step3: Convert the previous text to ascii code.
Step4: Convert the previous ascii code to binary data.
Step5: Reverse the previous binary data.
Step6: Gather each 8 bits from the previous binary data and obtain the ascii code from it.
Step7: Divide the previous ascii code by 4.
Step8: Obtain the ascii code of the previous result divide and put it as one character.
Step9: Obtain the remainder of the previous divide and put it as a second character.
Step10: Return encrypted text.

INPUT: Plaintext (StrValue), Key (StrKey).
OUTPUT: Ciphertext (EncryptedData).

1. Add the key to Text (StrKey + StrValue)---> full string (StrFullVlaue).
2. Convert the Previous Text(StrFullVlaue) to ascii code (hexdata).
3. Foreach (byte b in hexdata).
   a. Convert the Previous ascii code (hexdata) to binary data (StrChar).
   b. Switch (StrChar.Length).
      Case 7 ---> StrChar = "0" + StrChar.
      Case 6 ---> StrChar = "00" + StrChar.
      Case 5 ---> StrChar = "000" + StrChar.
      Case 4 ---> StrChar = "0000" + StrChar.
      Case 3 ---> StrChar = "00000" + StrChar.
      Case 2 ---> StrChar = "000000" + StrChar.
      Case 1 ---> StrChar = "0000000" + StrChar.
      Case 0 ---> StrChar = "00000000" + StrChar.
   c. StrEncrypt += StrChar. (where, StrEncrypt= "")
4. Reverse the Previous Binary Data(StrEncrypt).
5. For i from 0 to StrValue.Length do the following:
   a. if (binarybyte.Length == 8).
      i. Convert the binary data (StrEncrypt) to ascii code and,
      ii. Divide the ascii by 4 → the result(first character) and,
      iii. The remainder of the previous → second character.
6. Return (EncryptedData).

Fig. 1: REA-Encryption Algorithm

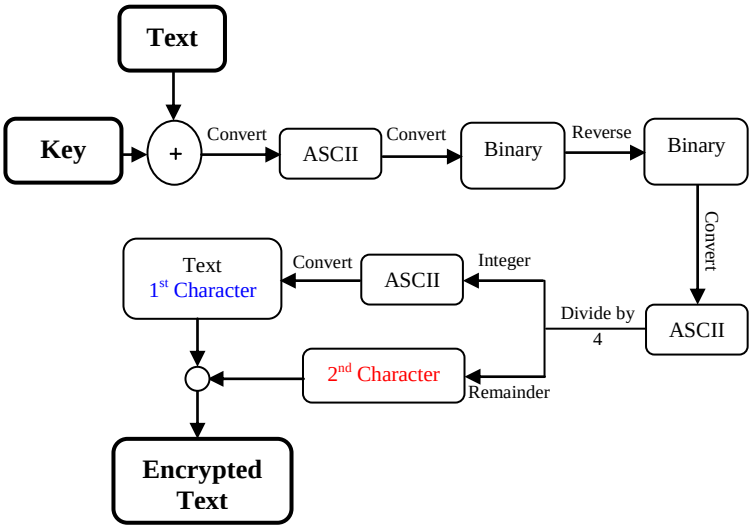

Fig. 2: Steps of the REA encryption algorithm

3.2 Decryption Algorithm of the REA

The steps of the decryption algorithm REA (Figure 3) are presented in the following steps (Figure 4):

Step1: Input the encrypted text and the key.
Step2: Loop on the encrypted text to obtain ascii code of characters and add the next character.
Step3: Multiply ascii code of the first character by 4.
Step4: Add the next digit (remainder) to the result multiplying operation.
Step5: Convert the previous ascii code to binary data.
Step6: Reverse the previous binary data.
Step7: Gather each 8 bits from the previous binary data and obtain the ascii code from it.
Step8: Convert the previous ascii code to text.
Step9: Remove the key from the text.
Step10: Return decrypted data.

INPUT: Ciphertext (EncryptedData), the Key (StrKey).
OUTPUT: Plaintext (DecryptedData),

1. For (i = 0; i < EncryptedData.Length; i += 2)
   a. Get the ascii code of the encrypted text
   b. newascii = (EncryptedData[i] * 4) + the next digit(remainder)[i+1].
2. Foreach (byte b in newascii).
   a. Convert the Previous ascii code (newascii) to binary data (StrChar).
   b. Switch (StrChar.Length).
      Case 7 ---> StrChar = "0" + StrChar.

Case 6 ---> StrChar = "00" + StrChar.
Case 5 ---> StrChar = "000" + StrChar.
Case 4 ---> StrChar = "0000" + StrChar.
Case 3 ---> StrChar = "00000" + StrChar.
Case 2 ---> StrChar = "000000" + StrChar.
Case 1 ---> StrChar = "0000000" + StrChar.
Case 0 ---> StrChar = "00000000" + StrChar.

c. StrDecrypt += StrChar.

3. Reverse the Previous Binary Data(StrDecrypt).
4. For i from 0 to StrDecrypt.Length do the following:
   a. if (binarybyte.Length == 8).
      i. Convert the binary data (StrChar) to ascii code (hexdata) and,
      ii. Convert the previous ascii code (hexdata) to the text (StrFullVlaue).
5. Remove the key from the text (StrFullVlaue - StrKey) → (StrValue).
6. Return (DecryptedData).

Fig. 3: REA-Decryption Algorithm

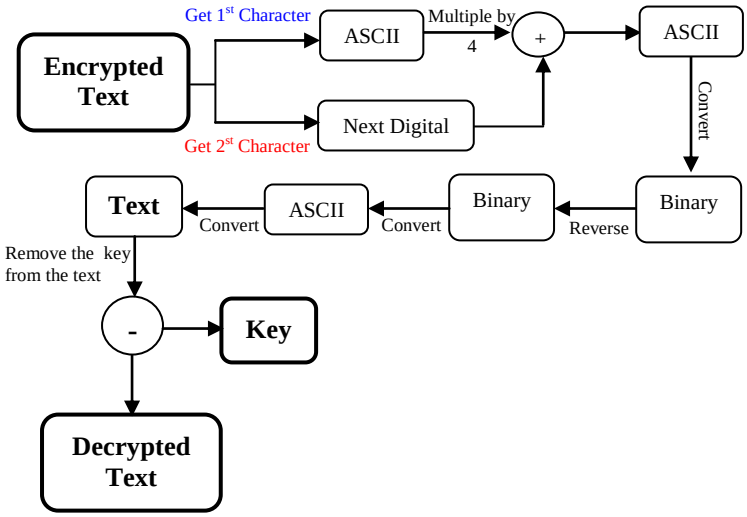

Fig. 4: Steps of the REA decryption algorithm

4. SECURITY AND PERFORMANCE FACTORS ANALYSIS

The following factors are used as the secure and efficient criteria [13] such as the *key space*, the *key sensitivity*, the *security of data against attacks*, the *computational speed*, the *information entropy*, and the *correlation coefficient*. Each of these factors is described in the following subsections.

Key Space Analysis

Key space is the total number of different keys that can be used in the cryptographic system. The security (strength) of the algorithm is a function of the length of the key. The longer the key, the more resistant the algorithm is to a successful brute-force attack. Key length is universally expressed as a number of bits [13]. A key length of *N*-bits has the key space 2^N possibilities. From the cryptography point of view, the size of the key space should not be smaller than 2^{100} to provide a high level of security [20]. The secret key of the proposed encryption algorithm REA is 256-bits long, can be increased, the key space is about 2^{256} ($1.16x10^{77}$) different combinations of the secret key. A long key space is sufficient for reliable practical usage.

Key Sensitivity Analysis

A good encryption should be sensitive to a small change in the secret keys [21]. The proposed encryption algorithm REA is sensitive to a tiny change in the secret keys. If change a little in the secret key then the decrypted data is not performing.

For example, If the plaintext is encrypted using the proposed algorithm REA with the secret key "fa32198cda3427da" as shown in Figure 10. Then, it decrypted with the correct secret keys, the plaintext obtained after decryption as shown in Figure 11. It is noted that the decrypted is exactly the same of the plaintext. Hence, say that the proposed algorithm REA can successfully encrypt and decrypt without any loss of data. From experiments, the secret key "fa32198cda3427d**A**" changed one character from small to capital as shown in Figure 5. Hence, say that the proposed encryption algorithm REA is highly sensitive to a small change in the secret keys.

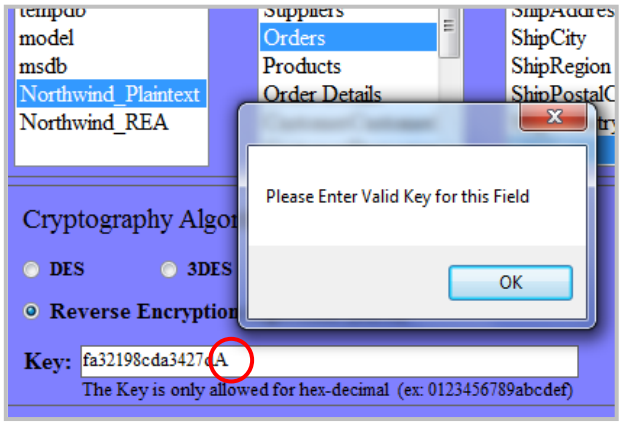

Fig. 5: Change one character in the secret key

Attack Analysis:

There are well known methods of attack, such as *brute-force*, which determines the number of *steps* and *time* required for a successful attack [4].

Attack Steps

Attack steps are defined as the number of steps required to perform the best known attack. The number of steps helps to determine the time that might be required for a successful attack using a particular processor, without having to actually run the attack on the algorithm. The proposed encryption algorithm REA using the key length 256-bits, can be increased, then the attack steps is about 2^{256} ($1.16x10^{77}$).

Attack Time

Attack time is defined as the time required performing the fastest known attack on a specified processor. For example, the machine that operates at 2000 (Mtops) times 60 (seconds/minute) times 60 (minutes/hour) times 24 (hours/day) 365 (days/year) equals $6.3072x10^{16}$ operations/year. The attack time in years were derived by dividing the attack steps by the pilot Mtops per year ($6.3072x10^{16}$ operations per year). Since proposed algorithm REA using 256-bits, then the attack time is about $1.839x10^{60}$ years.

Speed Analysis

It is an important tool to evaluate the efficiency of encryption algorithms is measuring the amount of time required to encrypt and decrypt process [9]. The encryption and

decryption times of the proposed encryption algorithm REA are given in the experimental results, it is fast enough.

Information Entropy Analysis

In the study of statistical analysis such as *Information Entropy* factor, it is used to measure the secure value of a cipher. The *entropy* of a message *m* of the size *n*, denoted *H(m)*, is the amount of information in the message [20-21].

The *entropy* of a given message *m* is defined by the weighted average:

$$H(m) = - \Sigma\{0 \le i \le n-1\}\ p(m_i)\ log_2\ p(m_i) \qquad (4.1)$$

Where $P(m_i)$ represents probability of m_i. If every symbol has an equal probability, i.e., $m=\{m_0,m_1,m_2,\ldots m_{255}\}$, then the result entropy is $H(m)=8$ which corresponds to an ideal value of entropy for message source *m*. Practically, the information entropies of encrypted data are less compared to the ideal case. To design a good data encryption scheme, the entropy of encrypted data close to the ideal case is expected.

A secure cryptosystem should be performing a condition on the information entropy that is the ciphertext should not provide any information about the plaintext. From the results, the entropy (secure) value of the proposed encryption algorithm REA is about 7.469.

4.6 Correlation Coefficient Analysis

Statistical analysis such as correlation coefficient factor is used to measure the relationship between two variables; the plaintext and its encryption. This factor demonstrates to what extent the proposed encryption algorithm strongly resists statistical attacks. Therefore, ciphertext must be completely different from the plaintext [21]. The correlation coefficient is measured by the following equation [13]:

$$CorrCoef(x,y) = \frac{\sum_{i=1}^{n}(x_i - \mu(x))(y_i - \mu(y))}{\sigma(x)\sigma(y)} \qquad (4.2)$$

where $\mu(x)$ and $\mu(y)$ are the respective *means* of *x* and *y*:

$$\mu(x) = \frac{1}{n}\sum_{i=1}^{n} x_i, \text{ and } \mu(y) = \frac{1}{n}\sum_{i=1}^{n} y_i \qquad (4.3)$$

x and *y* are variables of the plaintext and ciphertext.

and the terms in the denominators (It is called the *standard deviations* of *x* and *y*) are:

$$\sigma(x) = \sqrt{\sum_{i=1}^{N}(x_i - \mu(x))^2}, \text{ and}$$

$$\sigma(y) = \sqrt{\sum_{i=1}^{N}(y_i - \mu(y))^2} \qquad (4.4)$$

If the correlation coefficient equals *one*, that means the plaintext and its encryption is identical. If the correlation coefficient equals *zero*, that means the ciphertext is completely different from the plaintext (i.e. good encryption). If the correlation coefficient equals *minus one* that means the ciphertext is the negative of the plaintext (Figure 6). So, success of the encryption process means smaller values of the correlation coefficient. The experimental results, the correlation coefficient value of the proposed encryption algorithm REA is about 0.0872.

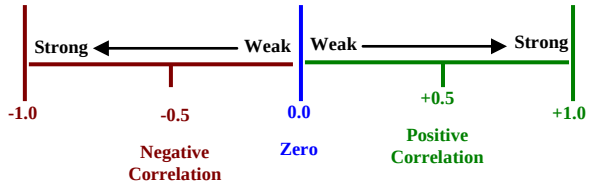

Fig. 6: Correlation coefficient shows strength and direction of correlation

5. EXPERIMENTAL RESULTS

A typical case study is studied in this section, to give the security and efficiency evaluation of a proposed encryption algorithm REA and to compare it with the most common encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish. The comparisons have been conducted for those encryption algorithms at computational speed (encryption and decryption time) and secure analysis such as information entropy and correlation coefficient. To make the encryption and decryption time is low as possible, cryptosystem should be optimized. Practically, the information entropies of encrypted data are less compared to the ideal case. Also, if the correlation coefficient equals zero, that means the ciphertext is completely different from the original.

All experiments were done on laptop IV 2.0 GHz Intel processor with 1 MB cache memory, 1 GB of memory, and one Disk drive 120 GB. The Operating System which was used is Microsoft Windows 7 professional. The results were executed based on the database Microsoft SQL Server 2005 is "*Northwind* ", which contains eight tables. The programming tasks were built by Microsoft Visual C# 2008. In the experiments, using two databases from the database "*Northwind*" are:

1. *Northwind_Plaintext* has not any encrypted fields. But it's used to encrypt and decrypt some fields (Table 1) by using the most common encryption algorithms namely: DES, 3DES, RC2, AES, Blowfish and the proposed algorithm REA.
2. *Northwind_REA* has the encrypted fields (Table 1) with using the proposed encryption algorithm REA (Figure 10).

Table1. Encrypted fields names

	Field Name	Table Name
F1	ContactName	Suppliers
F2	UnitPrice	Products
F3	ShipAddress	Orders
F4	Freight	Orders
F5	UnitPrice	Order Details
F6	Quantity	Order Details
F7	Description	Categories
F8	Notes	Employees
F9	ContactName	Customers
F10	ContactTitle	Customers

The keys used in the encryption data, kept secrecy in the encrypted table with the proposed encryption algorithm REA. This table contain five fields StrTable(is encrypted table name), StrField(is encrypted field name), StrKey(is encrypted key, it's used in encryption process), StrAlgo(is encryption algorithm name), and StrFieldType(is encrypted field type),

where the first three fields encrypted with proposed algorithm REA (Figure 7). Only the administrator will get these keys by using the password (Figure 8). After the administrator enters the password and selects the required database see the table of the encrypted keys in the database *"Northwind_REA"* (Figure 9).

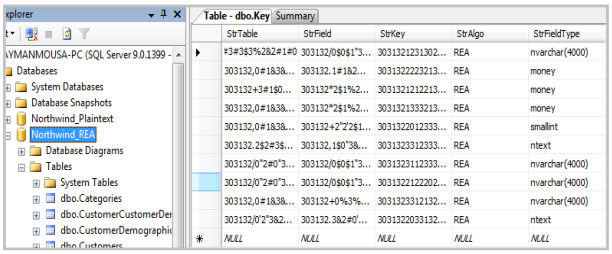

Fig. 7: Encrypted keys table with the proposed algorithm REA in the database *"Northwind_REA"*

Fig. 8: Login of the administrator to get the keys

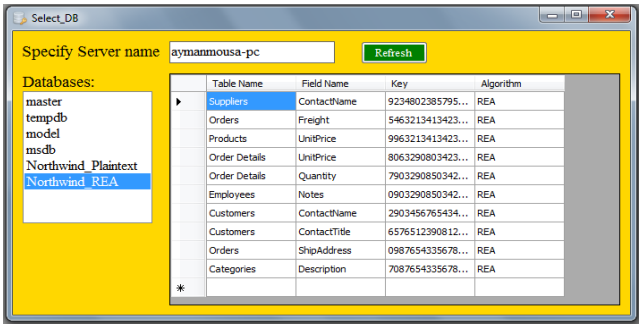

Fig. 9: Decrypted keys table with the proposed algorithm REA in the database *"Northwind_REA"*

The encryption and decryption time, the experiments were encrypted and decrypted ten different fields (shown in Table 1) with the proposed encryption algorithm REA and calculated execution time for each one. Then, calculated the average of the execution times for every encryption and decryption processes. Repeat these steps by other encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish (shown in Table 2 and 3). Figures 10 and 11 have shown once step from ten steps of using proposed encryption algorithm REA.

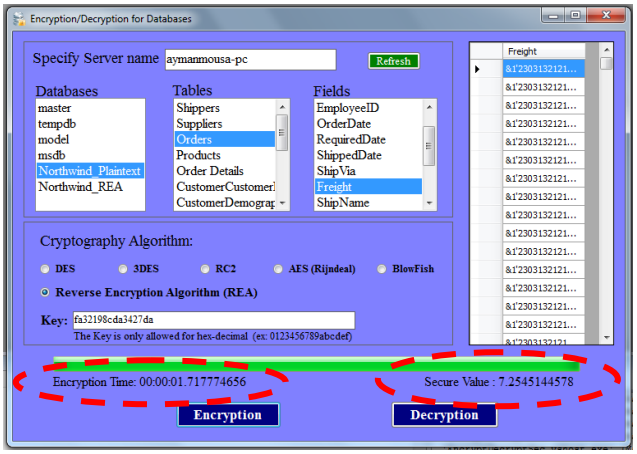

Fig. 10: Encrypted field with the proposed encryption algorithm REA

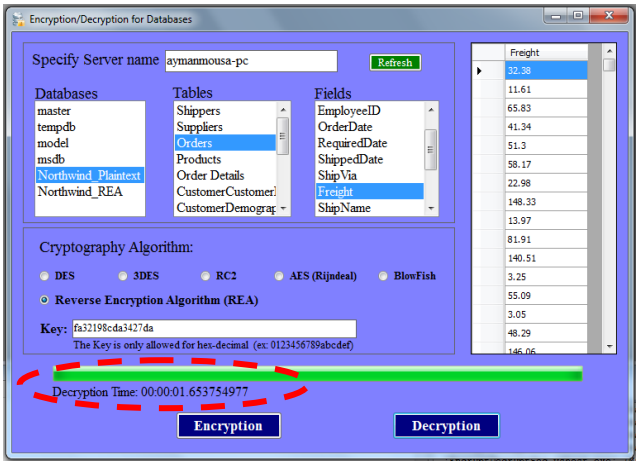

Fig. 11: Decrypted field with the proposed decryption algorithm REA

The results for the comparison are shown on Table 2 & Figure12 at the encryption time and Table 3 & Figure 13 at the decryption time. A first point; the results show the superiority of REA algorithm over other algorithms in terms of the encryption and decryption time. A second point; that Blowfish requires less encryption and decryption time than all algorithms except REA. A third point; that AES has an advantage over other 3DES, DES and RC2. A fourth point; that 3DES has low performance in terms of encryption and decryption time when compared with DES. It requires always more time than DES because of its triple phase encryption characteristics. A final point; it is found that RC2 has low performance in terms of encryption and decryption time when compared with other five algorithms.

Table 2. Comparative of encryption times (milliseconds) for encryption algorithms

	DES	3DES	RC2	AES	Blowfish	REA
F1	0.141	0.263	0.342	0.109	0.116	0.104
F2	0.359	0.419	0.395	0.329	0.296	0.266
F3	3.609	4.484	4.594	3.047	2.671	2.521
F4	4.063	4.469	4.513	3.297	2.544	1.718
F5	14.194	14.968	15.234	14.000	11.304	11.297
F6	15.906	17.547	17.328	15.484	12.452	12.360
F7	0.344	0.453	0.449	0.331	0.274	0.265
F8	2.960	3.203	3.531	2.688	2.051	2.005
F9	0.422	0.463	0.487	0.403	0.376	0.335
F10	0.421	0.438	0.442	0.386	0.346	0.334
Average Encryption Time	**4.242**	**4.671**	**4.732**	**4.007**	**3.243**	**3.121**

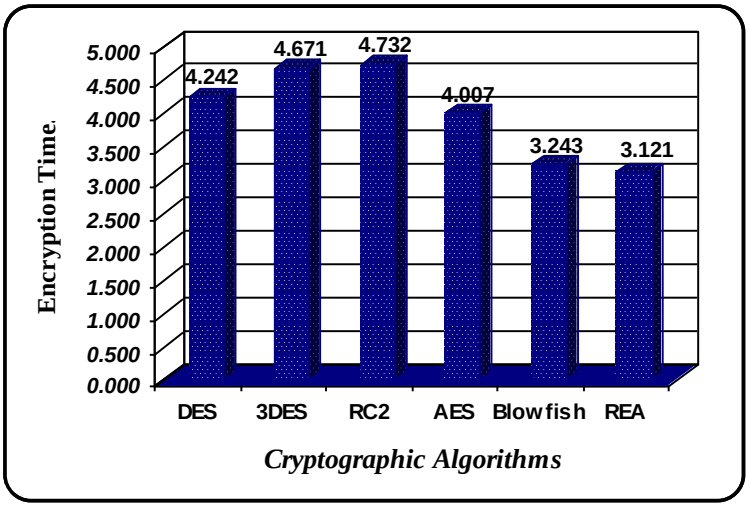

Fig. 12: Average of encryption times for encryption algorithms

Table 3. Comparative of decryption times (milliseconds) for decryption algorithms

	DES	3DES	RC2	AES	Blowfish	REA
F1	0.125	0.141	0.156	0.135	0.137	0.121
F2	0.343	0.359	0.384	0.344	0.322	0.271
F3	4.672	4.992	5.172	4.212	3.816	3.445
F4	4.313	4.625	4.516	4.103	3.417	1.654
F5	16.687	19.156	20.281	14.266	12.963	12.687
F6	17.797	20.313	21.406	15.125	13.761	11.030
F7	0.359	0.404	0.426	0.359	0.318	0.281
F8	3.312	3.891	3.906	3.319	2.175	2.743
F9	0.443	0.478	0.499	0.421	0.381	0.318
F10	0.438	0.447	0.456	0.398	0.315	0.308
Averages Decryption Time	**4.849**	**5.481**	**5.720**	**4.268**	**3.761**	**3.286**

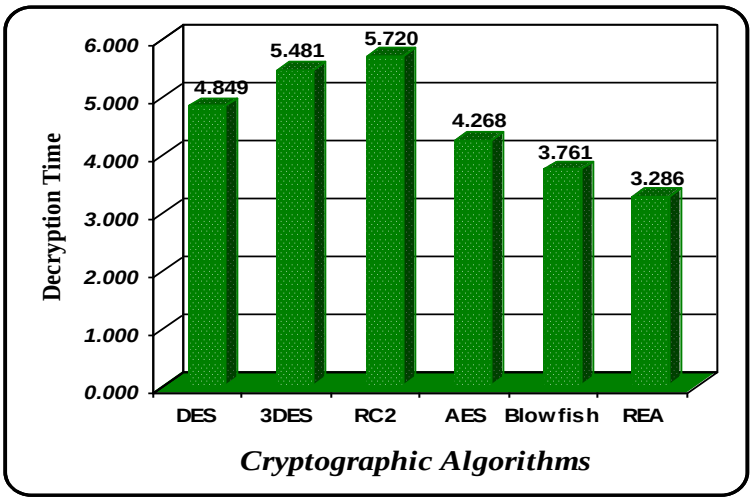

Fig. 13: Average of decryption times for decryption algorithms

Overall, the results showed that the proposed encryption algorithm REA has a very good performance compared to other encryption algorithms. Also, it showed that Blowfish and AES have a better performance than DES, 3DES, and RC2.

For the *information entropy factor (Security Analysis)*, a secure encryption algorithm should be performing a condition on the information entropy that is the ciphertext should not provide any information about the plaintext. The experiments results, the secure (entropy) values of encrypted fields with the proposed encryption algorithm REA (Figure 10) and comparing with the most common encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish. The results for this comparison are shown on Table 4, Figure 14 and Figure 15 at the secure (entropy) values.

Table 4. Comparative of secure (entropy) values for encryption algorithms

	DES	3DES	RC2	AES	BF	REA
F1	5.577	6.678	6.948	7.504	3.861	7.817
F2	4.765	5.872	6.354	7.415	3.801	7.181
F3	5.805	6.878	7.165	7.845	4.405	7.604
F4	4.748	5.847	6.451	7.423	3.791	7.255
F5	4.751	5.846	6.234	7.423	3.783	7.053
F6	4.769	5.841	6.587	7.656	3.799	7.154
F7	5.666	6.754	7.078	7.772	4.212	7.631
F8	5.258	6.339	6.975	7.583	4.163	7.735
F9	5.632	6.702	7.025	7.608	3.998	7.577
F10	5.623	6.667	6.952	7.801	4.358	7.684
Average Secure Value	**5.259**	**6.342**	**6.777**	**7.603**	**4.017**	**7.469**

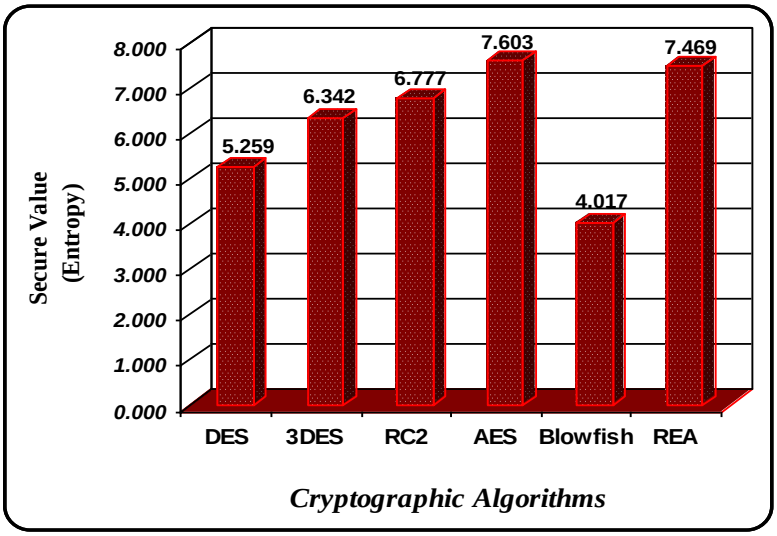

Fig. 14: Average of secure values for encryption algorithms

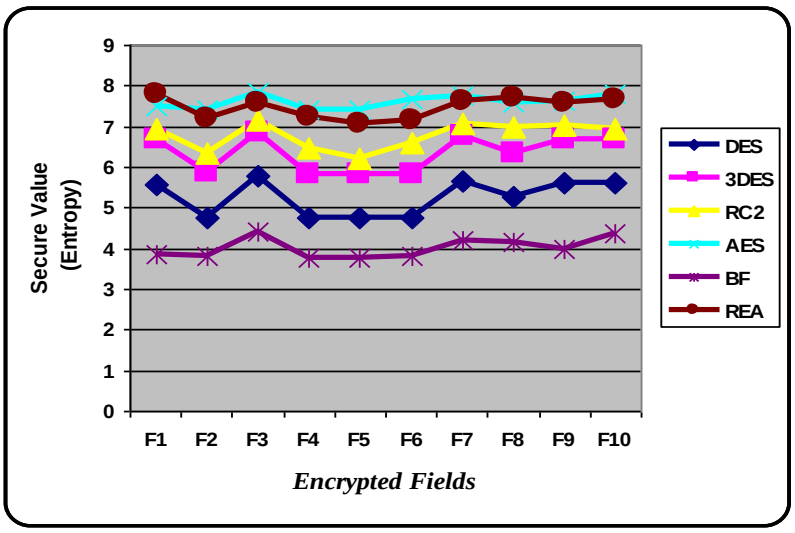

Fig. 15: Secure values of encrypted fields for encryption algorithms

The secure (entropy) of encrypted data by proposed encryption algorithm REA (is about 7.469) is a less compared to the ideal case. So, design REA is a security in data encryption.

The correlation coefficient factor (Security Analysis), it is used to measure the relationship between two variables; the plaintext and its encryption. Therefore, ciphertext must be completely different from the plaintext. Therefore, using two databases "*Northwint_Plaintext*" and "*Northwind_REA*" see figure 16. So, success of the encryption process means smaller values of the correlation coefficient. The results, the correlation coefficient values of encrypted fields by the proposed encryption algorithm REA are shown in table 5.

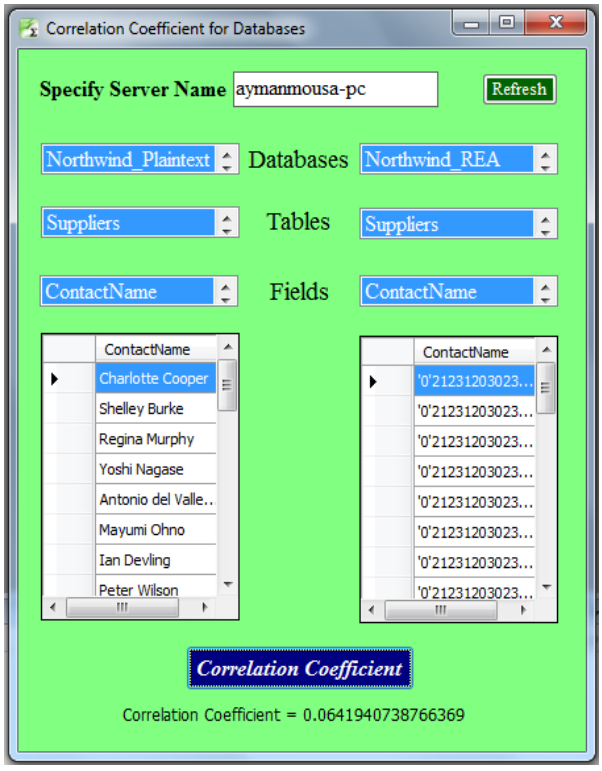

Fig. 16: Correlation coefficient value of encrypted field with proposed encryption algorithms REA

Table 5. Correlation coefficient values of encrypted fields with proposed encryption algorithms REA

	Correlation Coefficient Values
F1	0.0642
F2	0.1843
F3	0.0168
F4	0.0545
F5	0.2526
F6	0.0821
F7	0.0000
F8	0.0826
F9	0.0000
F10	0.1352
Average	**0.0872**

The correlation coefficient of encrypted data with the proposed algorithm REA (is about 0.0872) is a small value to compare zero value. So, the proposed algorithm REA is a good or secure encrypted data.

Finally, the encryption and decryption time results showed that the proposed encryption algorithm REA has a very good performance compared to other encryption algorithms. The security (information entropy) results show that the proposed encryption algorithm REA and AES have a better secure than DES, 3DES, RC2, and Blowfish. Also, the correlation coefficient results show that the proposed encryption algorithm REA has a strong security. Overall, the proposed encryption algorithm REA achieved balance between the security and efficiency in databases.

6. CONCLUSION AND FUTURE WORK

Encrypting sensitive data in the database becomes more and more crucial for protecting from being misused by intruders who bypass conventional access control mechanisms and have direct access to the database. Must be studied the performance and security of a new scheme in a systematic way. For this purpose, the proposed in this paper to address these issues and contributed in the following:

1. Introducing a novel encryption algorithm REA, restating its benefits and functions over other similar encryption algorithms. It is limiting the added time

cost for encryption and decryption so as not to degrade the performance of a database system.

2. Evaluating the efficiency of the proposed encryption algorithm REA and compares it with the most common encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish. It indicates the speed of encryption and decryption process. The results show the superiority of REA algorithm over other algorithms in terms of the encryption and decryption time.

3. Analyze the security of the proposed encryption algorithm REA and compares it with the most common encryption algorithms namely: DES, 3DES, RC2, AES and Blowfish. A comparison has been presented for the security (it is used to measure the information entropy). The results show, the secure (entropy) of encrypted data by proposed encryption algorithm REA (is about 7.469) is a less compared to the ideal case. So, designing REA algorithm is a strong security in databases encryption.

4. The correlation coefficient reflects the security measure as well. So, success of the encryption process means smaller values of the correlation coefficient. The results show, the correlation coefficient values of encrypted fields by proposed encryption algorithm REA (is about 0.0872).

In future work, interests in extending a novel encryption algorithm REA to support query processing performance on encrypted databases. In addition, we would like to extend and apply the proposed encryption algorithm REA in other kind of databases such as distributed DBMSs and object oriented DBMSs.

7. REFERENCES

[1] Bouganim L. , and Pucheral P., "Chip-secured data access: Confidential data on untrusted servers", *Proceedings of the 28th International Conference on Very Large Data Bases*, Hong Kong, China, pp.131-142, 2002.

[2] Castano S., Fugini M., Martella G., and Samarati P., "*Database Security*", Addison-Wesley, 1995.

[3] Chen G., Chen K., and Dong J., "A Database Encryption Scheme for Enhanced Security and Easy Sharing", *Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design*, 2006.

[4] Coppersmith D., "The Data Encryption Standard (DES) and Its Strength against Attacks", *IBM Journal of Research and Development*, pp.243-250, May 1994.

[5] Daemen, J., and Rijmen, V. "Rijndael: The Advanced Encryption Standard (AES)", *Dr. Dobb's Journal*, PP.137-139, March 2001.

[6] Damiani E., Vimercati S., and Foresti S., "Key Management for Multi-User Encrypted Databases", *Proceedings of ACM Storage SS'05*, pp.74–83, 2005.

[7] El-Fishawy N., "Quality of Encryption Measurement of Bitmap Images with RC6, MRC6, and Rijndael Block Cipher Algorithms", *Proceedings International Journal of Network Security*, PP.241–251, Nov. 2007.

[8] Ferraiolo D., and Kuhn R., "Role-based Access Controls", *Proceedings of NIST-NCSC'92*, pp.554–563, 2002.

[9] Idrus S., and Aljunid A., "Performance analysis of encryption algorithms text length size on web browsers," *IJCSNS International Journal of Computer Science and Network Security*, vol.8 no.1, pp. 20-25, Jan. 2008.

[10] Jingmin H., and Wang M., "Cryptography and Relational Database Management Systems", *Proceedings IEEE Symposium on the International Database Engineering & Applications*, Washington, USA, 2001.

[11] KIM Y., and Hong E., "A Study of UniSQL Encryption System: Case Study of Developing SAMS", *proceedings of ICACT 2007*, Volume III, pp. 577-582, Seoul, Korea, 2007.

[12] Mattsson U., "A Database Encryption Solution That Is Protecting against External and Internal Threats, and Meeting Regulatory Requirements: A practical implementation of field level privacy", 2005.

[13] Musheer A. and Shamsher M. "A New Algorithm of Encryption and Decryption of Images Using Chaotic Mapping ", International Journal on Computer Science and Engineering, Vol.2, No.1, pp.46-50, 2009.

[14] Results of Comparing Tens of Encryption Algorithms Using Different Settings-Crypto++ Benchmark, Retrieved Oct. 1, 2008. (http://www.eskimo.com/ weidai/benchmarks.html)

[15] Salama D., Abdual Kader H., and Hadhoud M., "Studying the Effects of Most Common Encryption Algorithms", *International Arab Journal of e-Technology*, Vol. 2, No. 1, pp.1-10 January 2011.

[16] Salama D., Abdual Kader H., and Hadhoud M., "Wireless Network Security Still Has no Clothes", *International Arab Journal of e-Technology*, Vol. 2, No. 2, pp.112-123, June 2011.

[17] Schneier B., "*Applied Cryptography Second Edition: protocols, algorithms, and source*", Beijing: China Machine Press, 2000.

[18] Schneier B., "The Blowfish Encryption Algorithm", *Retrieved*, 2008.

[19] Stallings W., "*Cryptography and Network Security Principles and Practice*" 4th Ed. Prentice-Hill Inc. 2005.

[20] Weerasinghe T., "Secrecy and Performance Analysis of Symmetric Key Encryption Algorithms ", *International Journal of Information & Network Security (IJINS)*, Vol.1, No.2, pp. 77-87, June 2012.

[21] Zeghid M., Machhout M., Khriji L., Baganne A., and Tourki R., "A Modified AES Based Algorithm for Image Encryption", *World Academy of Science, Engineering and Technology 27 2007*, pp. 206-211, March 2007.