IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555
Vol. 2, No.5, October 2012

A New Block Based Database Encryption Algorithm

[1]Anjana Nigam, [2]Prof. Roopali Soni
[1]M. Tech. Scholar (CSE) TCT, Bhopal, anjana.ngm@gmail.com
[2]M. Tech. Coordinator (CSE) TCT, Bhopal, rupal123_s@yahoo.com

Abstract: It is necessary to secure systems is well understood and securing data must be part of an overall security plan. Growing amounts of confidential data are being retained in databases and more of these databases are being made accessible via the Internet. As more data is made available electronically, it can be assumed that threats and vulnerabilities to the integrity of that data will increase as well. Database security is becoming an increasingly important topic and students need to develop core understandings in this area. The primary concern of database security is to prevent unauthorized access to data, prevent unauthorized tampering or modification of data, and to insure that data remains available when needed. The concepts related to database security are multifaceted. In this paper we are propose a new encryption/decryption block cipher algorithm. Presented results are showing that propose algorithm (PA) are better then as compare existing algorithm.

Key Word: ***Database, Encryption, Decryption, Security, Text, SQL, Key***

I. INTRODUCTION

The importance of security in database research has greatly increased over the years as most of critical functionality of the business and military enterprises became digitized. Database is an integral part of any information system and they often hold sensitive data. The security of the data depends on physical security, OS security and DBMS security. Database security can be compromised by obtaining sensitive data, changing data or degrading availability of the database. Though access control model were developed and found to ensure security, there were always chances of those access controls to be bypassed leading to a breach. To enforce the second layer of security, data being stored in the repository could be modified and stored in an encrypted format. This idea gave way to research in the design possibilities of databases. Two of such designs were Access Control Kernels and Encrypted Databases. Access Kernels were based on isolating and containing security policies inside separate modules. The downside of this design was that the value-dependent access restrictions were not possible. The cryptographic technique of using keys to encrypt and store data was applied to achieve security. There were many restrictions and challenges like operations\computations on encrypted data, view-based protection, etc. This research focuses on a security solution for protection of data at rest, specifically protection of data that resides in databases. Most databases are deployed and stored in some kind of a persistent storage device such as a disk. Periodically, even in-memory databases have the need to backup data; hence data could end up in a persistent storage device in plaintext. Many embedded devices that contain an embedded database hold sensitive data that must be protected. Encryption, the process of disguising data in such a way to hide its substance, is a very effective way to achieve security for data at rest.

Rests of the paper organized are as follow: Section II presents Proposed Work. Section III presents results analysis and finally Section IV presents conclusion.

II. PROPOSED WORK

Proposed Concepts: Proposed encryption is divided into two phases. And step of each phase is described one by one.

In phase1: 128 bits are taken on both sides. Then these 128bits are divided into 64bits on four parts. Now the first and last 64 bits are circular shifted. The first and third 64 bits are xored and the second and fourth bits are xored. Here the result forms 64 bits on both sides. Now, again these 64 bits are divided into 32 bits on four parts. The first and third 32 bits are interchanged their position on either sides and second and fourth interchange their position on its sides. The process of interchanging and circularly shifted on right and left side with xoring. At last, when 32 bits on both the side are combined then the concate process occurs and form the final result i.e. 128 bits cipher text. **In phase2**, the result of 128 bits is used as a left plaintext for this phase and remaining procedure is same as in phase1. In decryption, algorithm is also divided in two phases and step of each phase is described one by one. Here, also same procedure is follows as earlier done in encryption of phase1 and phase2.

Psudo Code for Proposed Encryption Algorithm

```
//string ProposedEncryptionAlgorithm (string
text, string keyy)
          // string key = "";
         //  string PlainText = "";
         //  string PT1 = "", PT2 = "";
          //     string CipherText = "";
          // key = keyy;
//Phase-I
Input 128 bits key and convert it into binary
value key = con_binary1(key);
Divide Key into K1 = key.Substring(0, 64)and
K2 = key.Substring(64, 64);
val1 = text;
while Loop (val1.Length % 16 != 0)
val1 += "Ã";
End While Loop
For Loop i = 0 To val1.Length
Select Plain Text of 128 bits from the
database PlainText = val1.Substring(i, 16);
```

```
Convertplain text into binary value PlainText
= con_binary1(PlainText);
Divide Plain Text into PT1 =
PlainText.Substring(0, 64) and      PT2 =
PlainText.Substring(64, 64);
Apply Right Circular Shift with 2 bits RK1 =
rightRotate(K1, 2);
Apply XOR between Key and Plain Text TempL =
xor(RK1, PT1);
ApplyLeft Cirucla Shift with 2 bits LP2 =
leftrotate(PT2, 2);
Apply XOR between Key and Plain Text TempR =
xor(K2, LP2);
Now Again Divide Left Value into TempL1 =
TempL.Substring(0, 32)and TempL2 =
TempL.Substring(32, 32);
Now Again Divide Right Value into TempR1 =
TempR.Substring(0, 32 and TempR2 =
TempR.Substring(32, 32);
Interchange these value with each other
t1 = TempL1;
TempL1 = TempL2;
TempL2 = TempR1;
TempR1 = TempR2;
TempR2 = t1;
Apply Right Circular Shift with three bits
TempL2 = rightRotate(TempL2, 3);
Apply Left Circular Shift with three bits
TempR2 = leftrotate(TempR2, 3);
Apply XOR TempR2 = xor(TempL1, TempR2);
Apply XOR TempL2 = xor(TempL2, TempR1);
Apply Right Circular Shift with Three Bits
TempL1 = rightRotate(TempL1, 3)
Apply Left Circular Shift with Three Bits
TempR1 = leftrotate(TempR1, 3);
Apply XOR TempL1 = xor(TempL1, TempR1);
Apply XOR TempL1 = xor(TempL1, TempL2);
Apply XOR TempR2 = xor(TempR1, TempR2);
//PHASE-II
TempL = TempL1 + TempL2;
TempR = TempR1 + TempR2;
Apply Right Circular Shift with 2 Bits RK2 =
rightRotate(K2, 2);
Appl XOR TempL = xor(RK2, TempL);
Apply Left Circular Shift with 2 Bits TempR =
leftrotate(TempR, 2);
Apply XOR TempR = xor(K1, TempR);
Now Divide Left Value n Sub Value TempL1 =
TempL.Substring(0, 32) and TempL2 =
TempL.Substring(32, 32);
Now Divide Right Value into Sub Vlaue TempR1
= TempR.Substring(0, 32) and TempR2 =
TempR.Substring(32, 32);
Interchnage These Value with Each Other
t1 = TempL1;
TempL1 = TempL2;
TempL2 = TempR1;
TempR1 = TempR2;
TempR2 = t1;
Apply Right Circular Shift with 3 Bits TempL2
= rightRotate(TempL2, 3);
Apply Left Circular Shift with 3 Bits TempR2
= leftrotate(TempR2, 3);
Apply XOR TempR2 = xor(TempL1, TempR2);
Apply XOR TempL2 = xor(TempL2, TempR1);
Apply Right Circular Shift with 3 Bits TempL1
= rightRotate(TempL1, 3);
Apply Left Circular Shift with 3 Bits TempR1
= leftrotate(TempR1, 3);
Apply XOR TempL1 = xor(TempL1, TempR1);
Apply XOR TempL1 = xor(TempL1, TempL2);
Apply XOR TempR2 = xor(TempR1, TempR2);
Now Combine All these Sub Value in one Value
CT = TempL1 + TempL2 + TempR1 + TempR2;
//CT = con_to_ascii1(CT);
Final Cipher TextCipherText += CT;
End Loop
```

Psudo Code for Proposed Decryption Algorithm

```
ProposedDecryptionAlgorithm(string text,
string Keyy) string key = "";
key = Keyy;
string PlainText = "";
string PT1 = "", PT2 = "";
string CipherText = "";

 //Phase-I
Input Key of 128 Bits and Convertits into
Binary Value key = con_binary1(key);
Divide Key int K1 = key.Substring(0, 64) and
K2 = key.Substring(64, 64);
string val1 = text;
while Loop val1.Length % 128 !=
val1 += "0";
End While Loop

For Loop i = 0 To val1.Length
CipherText = val1.Substring(i, 128);

Select Cipher Text of 128 Bits From The
Database
CipherText = con_binary1(CipherText);
Divide this Cipher Text into
TempL1 = CipherText.Substring(0, 32);
TempL2 = CipherText.Substring(32, 32);
TempR1 = CipherText.Substring(64, 32);
TempR2 = CipherText.Substring(96, 32);

Apply XOR TempL1 = xor(TempL1, TempL2);
Apply XOr TempR2 = xor(TempR1, TempR2);
Apply XOR TempL1 = xor(TempL1, TempR1);
Apply Right Reverse Circular Shift with 3
Bits TempR1 = rightRotate(TempR1, 3);
```

```
Apply Left Reverse Circular Shift with 3 Bits
TempL1 = leftrotate(TempL1, 3);
Apply XOR TempR2 = xor(TempL1, TempR2)
Apply XOR TempL2 = xor(TempL2, TempR1);
Apply Left Reverse Circular Shift with 3
TempL2 = leftrotate(TempL2, 3);
Apply Right Reverse Circular Shift with 3
Bits  TempR2 = rightRotate(TempR2, 3);
Interchange these Vlaue with Each Other
t1 = TempL1;
TempL1 = TempR2;
TempR2 = TempR1;
TempR1 = TempL2;
TempL2 = t1;
Combine Left sub Value into TempL = TempL1 +
TempL2;
Combine Right sub Value into TempR = TempR1 +
TempR2;

Apply XOR TempR = xor(K1, TempR);
Apply Right Reverse Circular Shift with 2
Bits  TempR = rightRotate(TempR, 2) and RK2 =
rightRotate(K2, 2);
Apply XOR TempL = xor(TempL, RK2);

TempL1 = TempL.Substring(0, 32);
TempL2 = TempL.Substring(32, 32);
TempR1 = TempR.Substring(0, 32);
TempR2 = TempR.Substring(32, 32);
//Phase II

Apply XOR TempL1 = xor(TempL1, TempL2);
Apply XOR TempR2 = xor(TempR1, TempR2);
Apply XOR TempL1 = xor(TempL1, TempR1);
Apply Right Reverse Circular Shift with 3
Bits  TempR1 = rightRotate(TempR1, 3);
Apply Left Reverse Circular Shift with 3 Bits
TempL1 = leftrotate(TempL1, 3);

Apply XOR TempR2 = xor(TempL1, TempR2);
Apply XOR TempL2 = xor(TempL2, TempR1);
Apply Left Reverse Circular Shift with 3 Bits
TempL2 = leftrotate(TempL2, 3);

Apply Right Reverse Circular Shift with 3
Bits  TempR2 = rightRotate(TempR2, 3);

Interchange These Sub Value with Each Other
t1 = TempL1;
TempL1 = TempR2;
TempR2 = TempR1;
TempR1 = TempL2;
TempL2 = t1;
Now combine Left Sub Value in TempL = TempL1
+ TempL2;
Now combine Right Sub Value in TempR = TempR1
+ TempR2;

Apply Right Reverse Circular Shift with 2
Bits  RK1 = rightRotate(K1, 2);
Apply XOR PT1 = xor(TempL, RK1);
Apply XOR PT2 = xor(K2, TempR);
Apply Right Reverse Circular Shift with 2
Bits  PT2 = rightRotate(PT2, 2);
Now CombinePlain Text and convert its into
orignal form PT = PT1 + PT2; PT =
con_to_ascii2(PT);
PlainText += PT;
End Loop
```

Brute force attack: Since longer symmetric keys require exponentially more work to brute force search, a sufficiently long symmetric key makes this line of attack impractical. With a key of length n bits, there are 2^n possible keys. This number grows very rapidly as n increases. Moore's law suggests that computing power doubles roughly every 18 to 24 months, but even this doubling effect leaves the larger symmetric key lengths currently considered acceptable well out of reach. The large number of operations (2^{128}) required to try all possible 128-bit keys is widely considered to be out of reach for conventional digital computing techniques for the foreseeable future [12]. Security level is the relative strength of an algorithm. An algorithm with a security level of x bits is stronger than one of y bits if x > y. The 128-bit security level is for sensitive information, and the 192-bit level is for information of higher importance [13]. Here proposed algorithm having 128 bits key length so there are 2^{128} possible keys. The larger number of operation (2^{128}) required to try all possible 128-bit keys is widely considered to be out of reach for conventional digital computing techniques for the future.

III. RESULT ANALYSIS

For this experiment, a laptop Pentium® Dual-Core CPU T4400 @2.20Ghz and 32-bit Operating System is used, in which performance data is collected. In the experiments, the laptop encrypts a database of different-different tuples and calculates encryption time and decryption time. Here some parameters are used for results analysis like average execution time for encrypt/decrypt the database which is shown in table 1, Average throughput which is show in table 2, CPU utilization for encryption/decryption which is shown in table 3 & 4 respectively. All the results are approximately calculated. Here the comparison between execution time, CUP Utilization, Throughput of encrypting/decrypting database on existing cryptographic algorithm with proposed cryptography algorithm is done. During processing, the content of the database and the key are both change simultaneously. One is the number of evaluated database and the size of evaluated database, where the number of evaluated database is the number of records that are stored in the database. For the experiment calculation we have select database which is shown in table 1.

Table 1: Database name with record Size

SNO.	Name of Database	Total Tusple Records	Size Of Database in

			Bytes
1	Academic Detail	709	68064
2	Climate Index	509	48480
3	Share Market	309	29690

The proposed system approximately 100 times has run. In each time, same database are respectively encrypted by AES and Proposed Algorithm (PA) by copying them. Finally a graphical representation for the table 1 is shown in graph 1 with blue line and orange line for encryption/decryption average execution time of existing system and proposed algorithm respectively. According to the graph, there is a tendency that average encryption /decryption execution time for proposed system and compared existing system increases with file size. But required execution time for the encryption/decryption through Proposed System is much smaller than encryption time for compared system. Another graphical representation for the table 2 is shown in graph 1 with blue line and orange line for encryption/decryption average throughput of existing system and proposed algorithm respectively. And finally graphical representation for the table 3 & 4 is shown in graph 3 & 4 with blue line and orange line for encryption/decryption CPU utilization of existing system and proposed algorithm respectively.

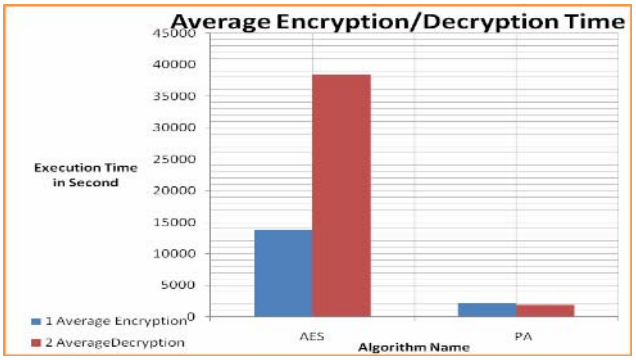
Average Execution Time for Encryption/Decryption: Execution time that is time taken to encrypt and decrypt the database. Average execution time is calculated as follow:

Average AES Execution Time = [(1199+12953+16565)/3]

Average PA Execution Time=[(1248+2532+2833)/3]

Table 1: Average Execution Time Comparison

SNO.	EXECUTION TIME	AES	PA
1	Average Encryption	13836	2204
2	Average Decryption	38393	1904



Graph 1: Average Encryption/Decryption Execution Tme Comparison

Average Throughput: With the help of execution time throughput can be calculated. It is the ratio of total size of database divide by total execution time during encryption/decryption. Throughput indicates the execution speed.

Throughput = Size of Database to be Encrypt/ Total Execution Time during Encryption

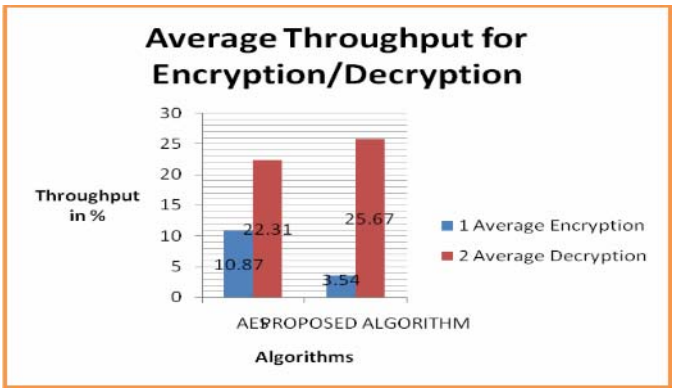
Average throughput is calculated as follow:

Average AES Throughput = [(24.76+3.742+4.108)/3]

Average PA Throughput=[(23.79+19.14+24.02)/3]

Table 2: Average Throughput Comparison

SNO.	THROUGHPUT	AES	PA
1	Average Encryption	10.87	3.54
2	Average Decryption	22.31	25.67

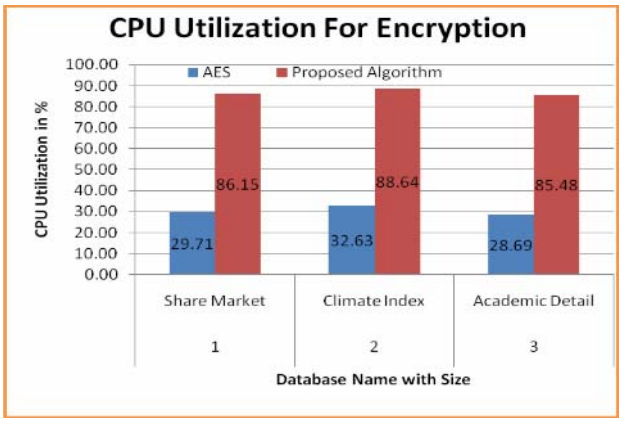


Graph 2: Average Encryption/Decryption Throughput Comparison

CPU Process Time: The CPU process time [4] is the time that a CPU is committed only to the particular process of calculations. It reflects the load of the CPU. The more CPU time is used in the encryption process, the higher is the load of the CPU. Proposed algorithm is taking large amount of time in execution then can say that CPU process time of proposed algorithm also good as compare existing algorithm. Table 3 is showing the CPU Utilization for encryption and table 4 is showing the CPU Utilization for decryption **.**

Table 3: CPU Utilization Comparison for Encryption

SNO.	DATABASE NAME	AES	PA
1	Share Market	29.71	86.15
2	Climate Index	32.63	88.64
3	Academic Detail	28.69	85.48

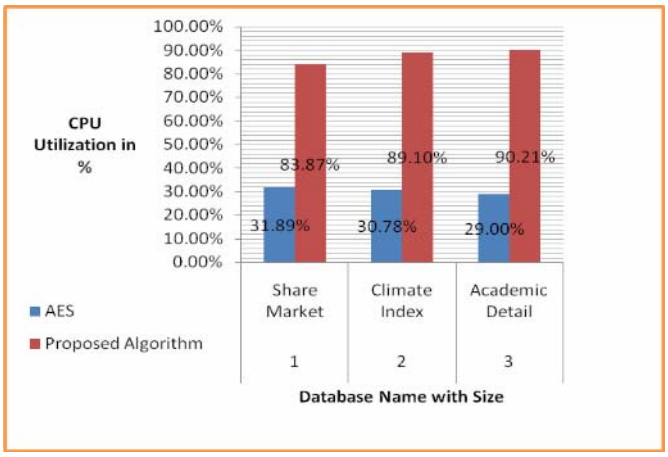


Graph 3: CPU Utilization Comparison for Encryption

CPU Utilization for Decryption:

Table 4: CPU Utilization Comparison for Decryption

SNO.	DATABASE NAME	AES	PA
1	Share Market	31.89%	83.87%
2	Climate Index	30.78%	89.10%
3	Academic Detail	29.00%	90.21%



Graph 4: CPU Utilization Comparison for Decryption

IV CONCLUSION

The number of existing system involving confidential information at the governmental, organizational and company levels is growing rapidly. Preserving data confidentiality, privacy and integrity in the semi-trusted database context, where the database is shared between many parties, is becoming one of the most challenging issues for the database community. The proposed work addresses this issue and contributes the following. It proposes a new cryptography algorithm for database based on data classification methods. Furthermore illustrates a query management system over an encryption database. Finally, it analyzes the execution time and security of data storage in the new cryptography framework. In Future work there are other important research issues related with research: first, improvement to design the best encryption algorithm for database on performance and security perspectives; second, access control methods use to control access for all parities using the database; and finally indexing and joining between different databases.

REFERENCES

[1] Maram Balajee "UNICODE and Colors Integration tool for Encryption and Decryption" published in International Journal on Computer Science and Engineering (IJCSE). ISSN : 0975-3397 Vol. 3 No. 3 Mar 2011

[2] A.Rathika, Parvathy Nair, M.Ramya "A High Throughput Algorithm for Data encryption" published in International Journal of Computer Applications (0975 – 8887) Volume 13– No.5, January 2011

[3] P. Wayner, Disappearing Cryptography : Information Hiding: Steganography and Watermarking. Morgan Kaufmann, 2nd edition, 2002.

[4] N. F. Johnson and S. Jajodia. Steganalysis of imges created using current steganography software. In IHW'98 – Proceedings of the International Information hiding workshop. April 1998.

[5] http://en.wikipedia.org

[6] http://www.joelonsoftware.com

[7] D. R. Stinson, "Cryptography Theory and Practice" CRC Press, Inc., 2002.

[8] IEEE Transactions on Circuits and Systems for Video Technology: Special Issue on Authentication, Copyright Protection, and Information Hiding, Vol. 13, No. 8, August 2003.

[9] Nadeem.A, "A performance comparison of data encryption algorithms," IEEE Information and CommunicationTechnologies, 2006

[10] PachghareV.K , Eastern Economy Edition, Cryptography and Information Security

[11] Stallings.W, *Cryptography and Network Security*, Prentice Hall, 4th Ed, 2005.

[12] Bruce Schneier "Applied Cryptography-Protocols, Algorithms and Source code in C".

[13] Menezes. A, Van.P, Oorschot, and Vanstone.S, "Handbook of Applied Cryptography", CRC Press, 1996.

[14] Yan Wang and Ming Hu "Timing evaluation of the known cryptographic algorithms "2009 International Conference on Computational Intelligence and Security 978-0-7695-3931-7/09 $26.00 © 2009 IEEE DOI 10.1109/CIS.2009.81

[15] "A Database encryption solution that is protecting against external and internal threats and meeting regulatory requirements" published in 2004

[16] Hasan Kadhem, Toshiyuki Amagasa and Hiroyuki Kitagawa "A Novel Framework for Database Security based on Mixed Cryptography" published in Fourth International Conference on Internet and Web Applications and Services 2009 IEEE.

[17] Dr. Anwar Pasha Abdul Gafoor Deshmukh and Dr. Riyazuddin QureshiTransparent "Data Encryption- Solution for Security of Database Contents" published in (IJACSA) International Journal of Advanced Computer Science and Applications,Vol. 2, No.3, March 2011.

[18] Indrani Balasundaram and E. Ramaraj "An Authentication Scheme for Preventing SQL Injection Attack Using Hybrid Encryption (PSQLIA-HBE)" published in European Journal of Scientific Research ISSN 1450-216X Vol.53 No.3 (2011), pp.359-368

[19] Oracle Security Handbook" by Theriault and Newman; Osborne/Oracle Press, 2001.

[20] Oracle Database Administration: The Essential Reference", Kreines and Laskey; O'Reilly, 1999.

[21] Pete Finnigan's Oracle Security web site, http://www.petefinnigan.com/orasec.htm

[22] Pressman, R., Software Engineering: A Practitioner's Approach, McGraw-Hill, 2000 (Chapter 10).

[23] Sommerville, I. and Sawyer, P., Requirements Engineering, Wiley, 1997.

[24] Diaa Salama Abdul Minaam, Hatem M. Abdual-Kader, and Mohiy Mohamed Hadhoud "Evaluating the Effects of Symmetric Cryptography Algorithms on Power Consumption for Different Data Types" International Journal of Network Security, Vol.11, No.2, PP.78-87, Sept. 2010

[25] McCarthy, Jack (April 3, 2000). "Governments Relax Encryption Regulations". PC World.

[26] http://www.cisco.com/web/about/security/intelligence/nextgen_crypto.html

Execution time comparison after Decryption with AES and PA