International Journal of Computer Trends and Technology
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V67I2P109

Volume 67 Issue 2, 60- 64, February 2019
© 2019 Seventh Sense Research Group®

Review Article

Comparison of Encryption Algorithms during Data Transmission at Rest and in Transit

P. Rajesh Kannan[1], R. Mala[2],

[1]*Research Scholar, Dept. Of Computer Science, MarudhuPandiyar College, Thanjavur, Tamilnadu, India.*
[2]*Assistant Professor, Dept. Of Computer Science, Alagappa University College of Arts and Science, Paramakudi, India*

***Abstract* -** *Securing sensitive data in databases has become very important nowadays because of the unencrypted form of unstructured data. Most open-source databases handle a huge amount of data in an unencrypted format accessible to anyone. Database operations such as read, edit, update, and delete are performed on the databases while in transit and rest. This paper proposes a secure encryption algorithm used to increase the packet delivery ratio and Throughput compared with the existing encryption algorithm used to encrypt the data at transit and rest. Compared with the existing algorithm, the average Throughput and packet delivery ratio results in a more secure algorithm with the validated results.*

***Keywords* -** *Encryption, data at transit, Throughput, packet delivery ratio, No SQL databases*

I. INTRODUCTION

Whether they handle the data in the industry, in business, or person wants to secure their sensitive data like adhar number, account numbers, PINs, etc. When databases store their unstructured data, they become larger due to the increasing demands for updated data to be maintained for future references. Nowadays, people move onto the NoSQL databases for the easy handling of the database operations such as read, update, edit and delete [1]. Due to the open nature of the NoSQL databases, anyone can view the details in the databases since they are not encrypted. Secure handling of private data has become very important nowadays in databases. Relational databases are securing their data with additional efforts provided along with the database. Data security is maintained by the Data Base Administrators (DBA) depending on the level of security needed by the concern. The huge storage of unstructured data on various mediums has become difficult to handle with the help of relational databases like SQL databases [2]. To handle these kinds of unstructured data, NoSQL(Not only SQL) databases are available as open-source databases like MongoDB, Cassandra, Redis, Hypertable, CouchDb, etc. Most open-source databases are not built with complete data security [3].

Hariharan et al. [4] discuss the various encryption techniques on the databases. This author surveyed different encryption methods like A Database Record Encryption Scheme Using the RSA Public Key Cryptosystem and its Master Keys, Chip-Secured Data Access: Confidential Data on Untrusted Servers, Fast and Secure Encryption for Indexing in a Column-Oriented DBMS, The Transport Layer Security (TLS) Protocol Version 1.2, etc. These suggested techniques differ based on their performance, access time, and key management [15].
In this paper, section ii elaborates on the related works by various authors in this field of encryption standards for database security. Section iii compares the various algorithms and methodologies used to encrypt the private data in databases to prevent unauthorized access to data. The proposed new algorithm called E-TDE is explained, and the improved packet delivery ratio is reported when data is at rest. Section IV explores the results and discussions based on the performance of the proposed E-TDE algorithm while used in the transit of data among the databases. Section v gives the conclusion based on the proposed methods.

II LITERATURE REVIEW

Data is to be protected when they are handled during the transit of the data, as well as they are at rest in the databases. Application-level security is to be adopted in some cases to secure the whole database carefully. Most the NoSQL databases do not provide encryption methods to protect the data. Databases contain all types of data to the user, whether sensitive or not.
For securing data in motion, all versions of MongoDB support TLS (Transport Layer Security) and SSL (Secure Socket Layer) to transfer the data over networks. This type of encryption technique is commonly used to secure website traffic and file sharing. While in transit, when the data travels from one point to another, it is unencrypted or 'in the clear .'MongoDB provides asymmetric key protocols to configure and secure the data in motion [12]. One of the challenges for MongoDB users is that when sensitive information is added to the database, users have to adopt a safe strategy of encrypting the

sensitive data in the MongoDB database with proper key management.

MongoDB Enterprise offers a storage-based file symmetric key encryption called Transparent Data Encryption (TDE) to encrypt the whole database files at the storage level. Version 3.2, MongoDB utilizes the Advanced Encryption Standard (AES) 256-bit encryption algorithm, an encryption cipher that uses the same secret key to encrypt and decrypt data [13]. But data at rest encryption is only available on MongoDB enterprise and atlas editions using the required Wired Tiger storage engine [14].

When TDE is used to encrypt the data, a unique, private key is generated. Each encrypted database file generates a new private symmetric key, and all keys in the storage device are encrypted using a master key. MongoDB never allows the master key to be stored on the same server as the encrypted data [5]. The security admin or the database must identify a secure storage location for the encryption key. MongoDB recommends third-party enterprise key management solutions; however, users can store the key locally using a key file. But according to the best practices, storing the key locally is risky and almost not recommended for key protection [6].

For securing data sensitive to the user and concern, they must be encrypted so that intruders or unknown persons do not intentionally tap them. The user adopts certain encryption methods to protect sensitive data when the data is at rest. When the data is transmitted from one storage area to the other, there is a need to protect the data during transit [7]. The basic operations such as read, write, append, update, edit and delete operations can be performed with the encryption method.

The data set is often the result of an individual's or organization's work, and protection of intellectual property rights becomes increasingly important. Watermarking and fingerprinting are some mechanisms to prevent unauthorized access to data sets [8]. Kurapati Sundar Teja et al. [9] explain the encryption-decryption methods, compare the performance with the popular encryption algorithms, and suggest a new design called FGPA.

III. COMPARISON OF ENCRYPTION METHODS

Among the various encryption methods suggested by various authors provide a few encryption methods to prevent unauthorized access to users' sensitive data [10]. Most databases do not provide encryption methods to protect the data set; there is a need to propose a standard procedure or encryption algorithms to access the data safely and securely [11]. The TDE algorithm is analyzed and compared with this paper's new proposed E-TDE algorithm. This proposed algorithm is used to encrypt the data, and the performance of the database is improved based on the packet delivery ratio and Throughput. This algorithm uses the RSA algorithm, one of the standard algorithms used to improve security for the database by providing encryption for the database operations during transit and at rest.

A. Encryption at rest

The comparison has been done with the existing TDE algorithm and proposed E-TDE. The following quantitative parameters are used to evaluate the performance of the E-TDE as follows:

B. Packet Delivery Ratio (PDR)

It defines the percentage of the total number of packets received at the destination and divided by the number of packets is sent by the source, as shown in Figure 1. Packet Delivery Ratio is calculated as follows:

$$\mathrm{PDR} = \frac{\text{No.of data packets received}}{\text{No.of data packets sent}} \times 100$$

Table 1. Packet Delivery Ratio- Documents inserted at rest

Number of Documents Inserted		5	15	25	35	50
No. Of Documents (in %)	TDE	87.6	87.9	90.1	91.2	93.4
	E-TDE	88.8	89.5	92.3	93.4	95.7

Figure 2 shows PDR when data is at rest. For a file size of 5MB, the TDE increases PDR by 87.6%, whereas the proposed E-TDE increases PDR by 88.8%. For a file size of 15MB, the TDE increases PDR by 87.9%, whereas the proposed E-TDE increases PDR by 89.5%. For a file size of 25MB, the TDE increases PDR by 90.1%, whereas E-TDE increases PDR by 92.3%. For a file size of 35MB, the TDE increases PDR by 91.2%, whereas E-TDE increases PDR by 93.4%. For a file size of 50MB, the TDE increases PDR by 93.4%, whereas E-TDE increases 95.7%.

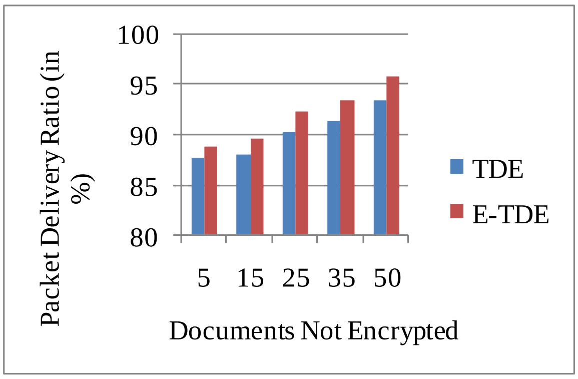

Fig. 2 Packet Delivery Ratio when data at rest

B. Encryption during transit of data

The data is transmitted among various sources and destinations during the usage of data extraction. The data manipulations such as insertion, edit, updation, and deletion of data are usually done on the data set during the transit. The packet delivery ratio is increased compared with TDE, which improves the performance of the database operations.

Table 2: Packet Delivery Ratio for Encryption at Transit

Number of Documents Inserted		5	15	25	35	50
No. of Documents (in %)	TDE	91.6	93.12	94.15	95.18	96.2
	E-TDE	92.09	93.78	94.66	96.96	98.72

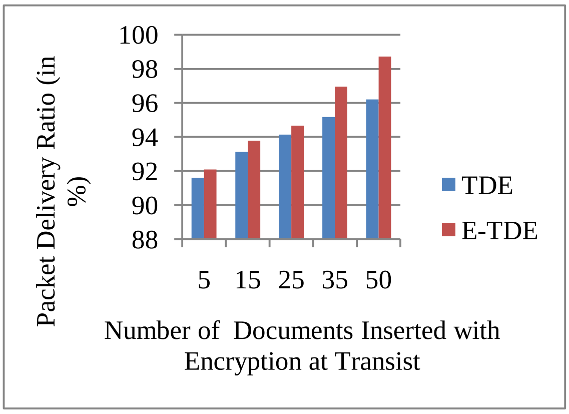

Fig. 3 Packet Delivery Ratio with Encryption at Transit

Figure 3 shows PDR with a different number of documents. For a file size of 5MB, the TDE increases PDR by 91.6%, whereas the proposed E-TDE increases PDR by 99.2%. For a file size of 15MB, the TDE increases PDR by 92.09%, whereas the proposed E-TDE increases PDR by 93.78%. For a file size of 25MB, the TDE increases PDR by 94.15%, whereas E-TDE increases PDR by 96.15%. For a file size of 35MB, the TDE increases PDR by 93.12%, whereas E-TDE increases PDR by 95.12%. For a file size of 50MB, the TDE increases PDR by 96.2%, whereas E-TDE increases 98.72%.

IV RESULTS AND DISCUSSION

The results of the application of the E-TDE algorithm are validated concerning two important parameters, namely, packet delivery ratio and Throughput. Compared to the TDE method, E-TDE shows significant improvement in the performance of the database operations when the data is at rest and in transit. This proposed encryption method improves the security of the database to a greater extent.

A. Packet Delivery Ratio

The proposed E-TDE method increases the Packet Delivery ratio when data is at rest. Figure 4 shows that the average PDR is increased by the proposed E-TDE concerning documents at the resting stage. When the file size increases, PDR also increases by the proposed E-TDE while documents are at rest.

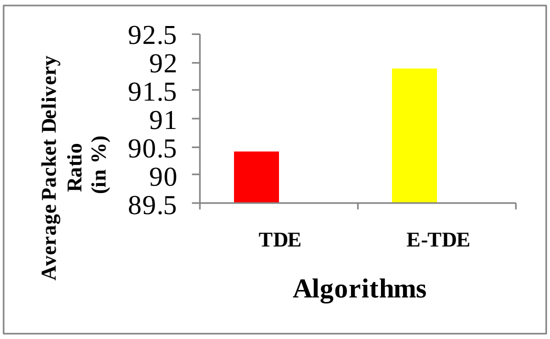

Fig. 4 Comparison of Average Packet Delivery Ratio with encryption at rest

Figure 5 shows that the average PDR is increased by the proposed E-TDE concerning the different number of documents. When the file size increases, PDR also increases by the proposed E-TDE. The results show that the proposed E-TDE significantly enhances PDR in the NoSQL database.

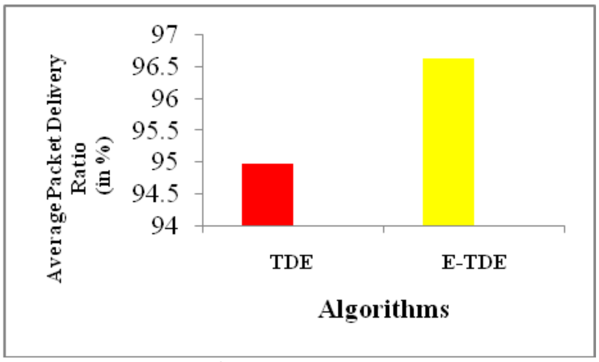

Fig. 5 Comparison of Average Packet Delivery Ratio with Encryption at Transit

B. Throughput

Throughput is one of the important parameters used to measure the performance of the database operations. It defines the average of successful message delivery over a communication channel. Throughput is calculated as follows:

Throughput = File Size / Transmission Time

1. Throughput - Documents at rest

Table 3. Throughput - Documents at rest

Number of Documents Inserted		5	15	25	35	50
Average Throughput (in Mbps)	TDE	18.84	10.14	9.30	6.5	5.4
	E-TDE	19.65	11.96	11.80	10.80	9.7

TDE produces 18.84Mbps Throughput, and the E-TDE produces 19.65Mbps Throughput for a 5MB file size. It is also simulated for 15MB, TDE provides 10.14 Mbps Throughput, the E-TDE 11.96Mbps, for 25MB, the TDE produces 9.30Mbps Throughput, and the E-TDE produces 11.80 Mbps throughput. For a file size of 35MB, the TDE increases Throughput by 6.5Mbps, whereas E-TDE increases throughput by 10.80%. For a file size of 50MB, the TDE increases Throughput by 5.4Mbps, whereas E-TDE increases throughput by 9.7%.

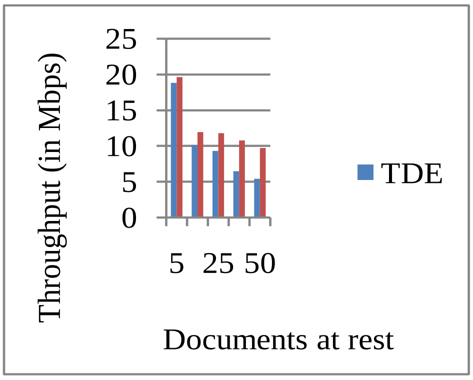

Fig. 6 Throughput with data at rest

Figure 7 shows that the average Throughput is increased by the proposed E-TDE concerning documents not encrypted stage. When the file size increases, Throughput increases by the proposed E-TDE while documents are not encrypted.

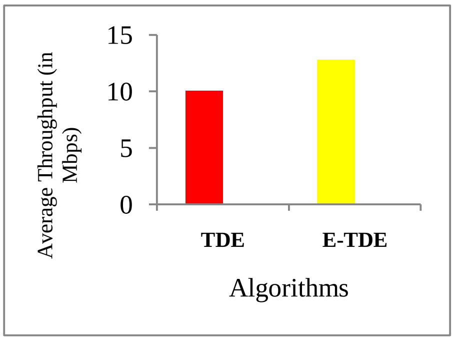

Fig. 7 Comparison of Average Throughput with data at rest

2. Throughput - Documents at transit

Table 4. Documents Encryption at Transit for Throughput

Number of Documents Inserted		5	15	25	35	50
Average Throughput (in Mbps)	TDE	20.64	14.12	15.25	12.5	9.5
	E-TDE	24.75	16.14	18.38	19.47	12.14

TDE produces 20.64Mbps Throughput, and the E-TDE produces 24.75Mbps Throughput for a 5MB file size. It is also simulated for 15MB, TDE provides 14.12 Mbps Throughput, the E-TDE 16.14Mbps, for 25MB, the TDE produces 15.25Mbps Throughput, and E-TDE produces 18.38 Mbps throughput. For a file size of 35MB, the TDE increases Throughput by 12.5Mbps, whereas E-TDE increases throughput by 19.47%. For a file size of 50MB, the TDE increases Throughput by 9.5Mbps, whereas E-TDE increases throughput by 12.14%.

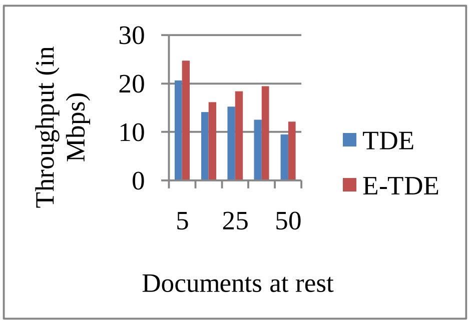

Fig. 8 Throughput with Encryption at Transit

Figure 9 shows that the average Throughput is increased by the proposed E-TDE concerning documents encryption at the transited stage. When the file size increases, Throughput also increases by the proposed E-TDE while document encryption is at the transited stage.

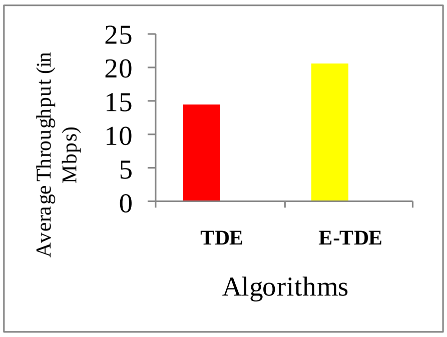

Fig. 9 Comparison of Average Throughput with Encryption at Transit

V. CONCLUSION

Encryption is one of the important techniques used to protect individual or enterprise data. Since most of the NoSQL databases do not provide adequate security in protecting the user's private data, intruders easily get access to the private data. There is a serious demand for secure access to data in databases provided with encryption methods. This paper proposes one encryption algorithm called E-TDE that is used to improve the performance of the database. This significant approach increases the total security of the user's valuable data from unauthorized users. The packet delivery ratio and Throughput are improved compared with the TDE algorithm, and the results are validated and reported. In the future, various combined methodologies can be adapted to improve the secure access of users' private data.

REFERENCES

[1] Jef Van Loon, Prof. Dr. C-C. Kanne, Ch. Sturm, "Database Security - Concepts, Approaches," **Article** in IEEE Transactions on Dependable and Secure Computing · Seminar in Database Systems, University of Zurich,

Department of Informatics, Autumn Term 2008, DOI: 10.1109/TDSC.2005.9 · Source: IEEE Xplore.

[2] Mubina Malik and Trisha Patel, "DATABASE SECURITY - ATTACKS AND CONTROL METHODS," International Journal of Information Sciences and Techniques (IJIST) Vol.6, No.1/2, March 2016.

[3] P. Rajesh kannan^1, R. Mala^2, “comparison of encryption algorithms on NoSQL databases," International Journal of Computer Sciences and Engineering, Vol.-6, Issue-10, Oct 2018.

[4] P.R.Hariharan & Dr. K.P. Thooyamani, Various Schemes for Database Encryption - A Survey", International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 19 (2017) pp. 8763-8767, Research India Publications.

[5] https://docs.mongodb.com/manual/core/security-encryption-at-rest.

[6] Suna Yin, Dehua Chen, Jiajin Le, China, 2016 IEEE, "STNOSQL Creating NOSQL Database on the Sensible Things Platform.

[7] E. Bertino and R. Sandhu. Database security - concepts, approaches, and challenges.Dependable and Secure Computing, IEEE Transactions on, 2(1):2–19, March 2005.

[8] Shelly Rohilla, Pradeep Kumar Mittal, Database Security: Threats and Challenges, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 5, May 2013.

[9] Kurapati Sundar Teja et al. “ Data Encryption and Decryption Algorithm Using Hamming Code and Arithmetic Operations," Int. Journal of Engineering Research and Applications Vol. 5, Issue 8, (Part - 1) August 2015, pp.81-82

[10] Vinod Shokeen, Niranjan Yadav, "Encryption and Decryption Technique for Message Communication," International Journal of Electronics & Communication Technology, Vol. 2, Issue 2, June 2011, pp. 80-83.

[11] Obaida Mohammad Awad Al-Hazaimeh, "A New Approach For Complex Encrypting and Decrypting Data," International Journal Of Computer Networks & Communications (IJCNC) Vol.5, No.2, March 2013, pp. 95-103.

[12] D. Kulkarni. A fine-grained access control model for key-value systems. Proceedings of the third ACM conference on data and application security and privacy pages 161–164. ACM, 2013

[13] P. Colombo and E. Ferrari. Enhancing MongoDB with Purpose Based Access Control. In IEEE Transactions on Dependable and Secure Computing, November 2015 DOI:10.1109/TDSC.2015.2497680

[14] Boyu Hou, Kai Qian, Lei Li, Yong Shi, Lixin Tao, Jigang Liu, USA, 2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing, "Mongo Database NOSQL Injection Analysis and Detection."

[15] Anil Kumar, Harsha H L, B. Swaroop Reddy, K.Sunil Kumar Reddy, Krishna N, "Homomorphic Encrypted MongoDB for Users Data Security," International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 5, Issue 6, June 2018