Journal of Computer and Mathematical Sciences, *Vol.10(1),190-196 January 2019* *ISSN 0976-5727 (Print)*
(An International Research Journal), *www.compmath-journal.org* *ISSN 2319-8133 (Online)*

Protection of Data Stored in Transparent Database System using Encryption

Arvind Kumar Maurya[1], Avinash Singh[1], Unnati Dubey[2], Shivansh Pandey[3] and Upendra Nath Tripathi[*]

[1*]Department of Computer Science,
D.D.U. Gorakhpur University, Gorakhpur (U.P.) -273009, INDIA.
[2]Department of Computer Science,
Marwad Business School, Gorakhpur (U.P.), INDIA.
[3]Department of Information Technology,
J.S.S. Academy of Technical Education, Noida (U.P.), INDIA.
***** Corresponding Author

(Received on: January 7, 2019)

ABSTRACT

In transparent database data are stored at different severs which is easily visible and accessible to every user. End users usually access information through applications that intermediate between the user and the database. These applications provide interface to access the database system i.e. transparent to end users. This is expected and even logical aspect of the nature of most database management systems (DBMS). There is always possibility of data loss and data theft by an unauthorized user. Now-a-days research on Distributed database management system (DDBMS) found to best match to handling the transparent type database storage and processing rules via communication links.

In this paper, we have introduced a method of encrypting the transparent database files and data. The prime purpose here is to prevent an unauthorized access to the data by restoring the files from another server. These physical files include the database file (.mdf), the transaction log file (.ldf) and the backup files (.bak).

Keywords: Database, server, security, encryption.

1. INTRODUCTION

The different business houses having some of their business policy; these companies and business houses opted the transparent database system to store their data at several server

sites in transparent fashion[1]. The data stored concept in data warehouses generally fallows the transparent database storage format. These storage sites/servers doses not have any protection technique to save guard these transparent data; one could easily reach and access these data files by their some business rule or procedures. Most of the transparent database system the data is stored on different servers along with other company data residing with them and which are visible to every end user. Some time there is maximum possibility of malicious modification and theft of these valuable data by any non authentic user. Therefore, there we need to protect these data items data structure and stored procedures that are stored in transparent database[2,3,4].

A distributed database management system is a collection of databases model in which database is logically fragmented and these fragments are distributed and stored locally or remotely fashion way on different machines or sites and these fragments interact with each other or get process through some communication links[5,14]. A distributed type database management system is a huge collection of database size and having logical fragmentation of different sizes also not even use same database model. Such fragmented database tables are get replicated and stored over multiple database sites, located at different geographical locations and these database sites interact logically or remotely with each other via some communication links in such way that its internal details of distribution are totally hidden or being transparent form end users. The Distributed database management system (DDBMS) concept permits the management of the distributed database fragments and makes the distribution transparent to the user[6,7].

Here, we propose a noble technique to protect the data of transparent database by managing through distributed database system concept and further by encrypting each replicating fragments in the physical files rather than data which are stored in those servers and sites. When these stored files get encrypted the associated data, structures and procedures maintained in also being encrypted[8].

When any data or structure got to save in distributed database file system and then these files get encrypted and finally physically uploaded for store at servers. For retrieving the data elements it is needed to first access and then decrypts the file that owns data and then decrypted files data is made available to end user. In this technique data and structures are automatically encrypted when it is written to disk and automatically decrypted when accessed through their application programs. The data distribution and key management is built-in, eliminating the complex task of creating, managing logical fragmentation, replication and securing all these through encryption key system[9,10].

2. LITERATURE REVIEW

Many studies have been published on attempts of improving the performance of Transparent Database System. These researches have mostly investigated the site-independent schemata based system for designing secure database model. Such site-independent systems are defined using the same data model as the local DBMSs[11]. In this section, we present the main contributions related to encryption approach for protection of data in transparent database.

The authors of present a new formulation for the problem of protection of data with database encryption keys (DEKs) which are used to encrypt columns and table spaces within the databases minimum cost for both structured and unstructured data, by grouping sites which are nearer to each other into one cluster, hence they have low cost[12,13]. Also, a dynamic clustering method is adopted for both structured and unstructured database to reduce the movement of data between sites. In Transparent Database Encryption (TDE), author facilitates to protect data by encrypting the physical files of the database[14]. Encryption as a Service (EaaS) provided by HashiCorp's Vault to assist security with transit service in which databases never store plaintext data. For maximum security, the user or application does not need to manage TDE master encryption keys rather by combining TDE and Vault EaaS so they can adopt modern security best practices[2,10]. The paper published in the proceedings of the IEEE is publicly available algorithm that stores locally encrypted files and sends messages and documents securely to the recipients. Such security algorithm uses a mix of technologies and services with respect to security, costs and user "friendliness"[15].

3. ARCHITECTURE AND DESIGN OF TRANSPARENT DISTRIBUTED DATABASE ENCRYPTION TECHNIQUE

The distributed database system found model in ability to deal with transparent database system concept in storage and associated business rules.

This Transparent Distributed Database Encryption Technique (TDDET) operates on real-time input /output encryption and decryption of the data and associated log files when these files either written or reads form media or even when transmitted via communication links.

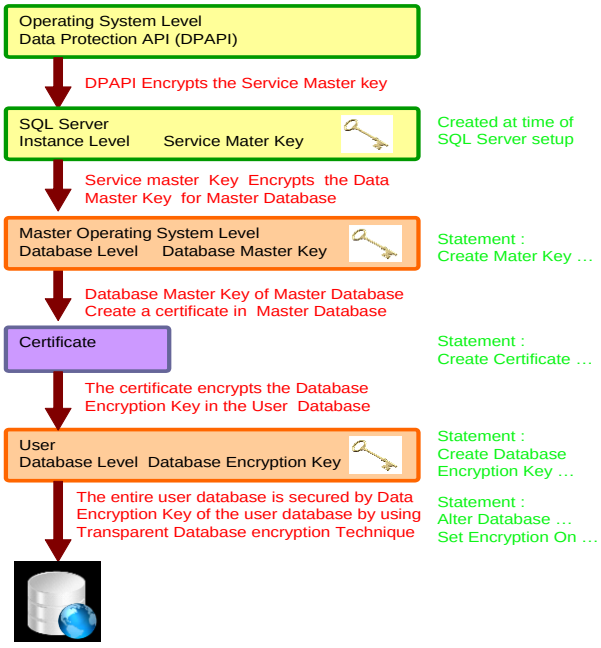

Fig. 1 Architecture and Design of Transparent Distributed Database Encryption Technique

The encryption process uses a database encryption key (DEK), which is stored in the database boot record for availability during recovery or installation time. The (DEK) is a symmetric key secured and authenticated by using time stamped based certificate which stored in the master database of the server. This certificate is itself stored in encrypted form by asymmetric key protected method using an encryption key module (EKM). TDDET protects data in physically stored the data and log files at storage media. TDDET has involvement to encrypt decrypt the data while transmission through communication links for updating or processing. It provides the ability to complement with many business rules, laws, regulations, and guidelines established in various industries. This enables application users to model data in distributed fashion and encrypts them by using AES and 3DES encryption algorithms without any changing for existing applications.

The server, database, data mining tools, federal and securities agencies network links share common boundary separated by a firewall with outside network. The user or authorities interacts with database servers though a web portal and some communication link provide by the site. The information retrieval or being transferred in encrypted form by neural network, AES, 3DES, RC5 and hashed by MD5, SHA-1 algorithms. In the context of tackling data and information theft found to quiet successful without effecting easiness and transparency of data.

4. IMPLEMENTATION OF TRANSPARENT DISTRIBUTED DATABASE SYSTEM ENCRYPTION TECHNIQUE

The transparent database encryption technique implementation process has to fallow six steps.

1. Transformation to Distributed Database System

Any chosen database may first transform to distributed database system then further encryptions and security parameters would be implemented.

2. Generation of Master Service Key

First master service key is generated with help of master service serving OS on which system it reside first and interact with database during first time installation and datatabase stored in as OS file (system software wallet) or separately attached managed by a Extended Hardware Security Module (EHSM), externally from database.

3. Generation of Database Secret Master Key

A single database Secret Master Key is generated at first installation time of database. For security reason the secret master key stored in encrypted form in separate file create by security and coordinating (DDBA). By maintaining dependency on single instance of database secret master key for the users interested in accessing the instance of databases have to fully depending upon the master key and having needs to share their secret key with this database secret mater key.

4. Generation Certificate protected by the Master Key

A certificate having timestamp is generated and protected through encryption by involvement through use of Secret Master Key. The use of this certificate is for generation and verification of some confidential Encryption Key utilize for building trust in between user and database.

5. Maintaining of Backup copies of the Certificates

A backup copy of certificate is also created and maintained for accidental loss or damage of original certificate. The old and new certificate copies are stored along the timestamp validity period for backup or recovery purpose.

6. Generation of Database Encryption Key and protect them by the Certificate

Finally a Database Encryption Key is generated and secured them by encrypting it with help of the Certificate. Each database access, transaction and processing session are secured and managed through these keys.

7. Turning Encryption On

By turning on the encryption the data file of user could be encrypted and secured for any unauthorized access or theft.

The dependency upon the encryption key hierarchy authorities in the *Master* database or coordinator, as well as maintaining their instance, prevents the database files from being restored to an instance of DBMS Server that does not contain the referenced keys. This level of protection and security polices provides a great ease of comfort in case of when any backup media containing our database backup files or stolen copy if fall into the wrong hands, unable to fetch any information.

We have maintained only one master key password. So at only first time we encryption database master key password field and the certificate backup password field in the form is enabled. These passwords are stored in encrypted form when need it decrypted then used and it don't transmitted over link openly but in encrypted format. In case of possibility of loss of Secret Master Key and Certificate a back up of master key and certificate is also created only once. Stored procedure is also created internally only owns under the master database of local server or coordinator. In any issue arises all participant server key certificate copy are asked via link and matched with other if site having different is reported disproved from participation and informed about others.

5. SECURITY PERFORMANCE ANALYSIS

1. Fully Master Key Interdependency

The process of implementing TDDET involves the creation of a database master key and certificate, asymmetric key for the Master database or coordinator. Only one database master key could be created for a given database so any other user databases that share the instance, and must have TDDET implementation to share a dependency upon the Master

database master key. This interdependency increases the importance of performing a backup of the Master database master key to ensure the continued accessibility of the TDET enabled databases.

2. Backup and Recovery

Backup files of databases that have TDDET enabled are also encrypted by using the database master encryption key. As a result, when we restore these backups, the certificate protecting the database encryption key must be available. This means that in addition to backing up the database, we have to keep them along with database. This make it sure that our maintain backups of the server certificates is to prevent any data loss. Data loss will result if the certificate is no longer available. The encrypted form of key and certificate ensures secrecy of their confidentiality haven't been broken.

3. Performance

Once the data files of distributed database is encrypted, any reference key or use of data by other databases, either they are TDDET enabled or not, because it requires encryption and decryption process to use them. While getting encrypted and decrypted these database files remains transparent to its users, and doesn't involve traffic or delay has very minimal performance impact on the entire instance.

Data remains encrypted in each logical fragments, temp files, cursors and work tables for spooling or transmission, undo segments and buffer cache, no residual data are left in the clear on disk .Encryption occurs on table creation and distribution but each stage data always remains encrypted. It also provides commit and the ability to rollback transactions in any failure case. Data at any stage is automatically encrypted when it is written on disk and automatically decrypted when accessed through the application same happens when data get permission to transmit over links.

6. CONCLUSIONS

This encryption method provides higher level of security to data through strong encryption and decryption technique used employed for encrypting the data files, log files and backup files without dissolving transparency property for its user. This method performs real-time I/O or transmission over links through encryption and decryption of the data and log files. It has distribution and encryption offers complete protection from accident loss of data. There is several possibilities to raise the security level by using stronger encryption algorithms based method employed with one way hashing techniques and implementing up to its cell level of transparent database.

REFERENCES

1. Sandy Steier, "Design Considerations for Transparent Databases", 1010 data, Inc. Content Available at site: www. 1010data.com January, (2010).
2. Oracle Financial Service, "Identity and Access Management Overview", Annual Report, 2014-2015, New Generation Consultants, *Oracle certified partner*, Inc Friday, Feb 27th, 2008 3.00 PM – 3.40 PM CST.

3. Solution Brief, "HSM & Oracle Transparent Database Encryption", Solution BRIEF.
4. John Magnabosco, "Transparent Data Encryption", Article, Redgate hub, 16 March (2010).
5. Bell, David and Jane Grisom, Distributed Database Systems. Workinham, England: Addison Wesley, (1992).
6. İlker Köse, GYTE, Veri ve Ağ Güvenliği, Distributed Database Security, Spring 2002.
7. Stefano Ceri, Giuseppe Pelagatti: Distributed Databases: Principles and Systems. McGraw-Hill Book Company, ISBN 0-07-010829-3 (1984).
8. Anwar Pasha Abdul Gafoor Deshmukh, Riyazuddin Qureshi, "Transparent Data Encryption- Solution for Security of Database Contents", *(IJACSA) International Journal of Advanced Computer cience and Applications,Vol. 2, No.3, March 201, page 25.* http://ijacsa.thesai.org/.
9. Understanding Transparent Data Encryption, *Microsoft*, MSDN.
10. Transparent Data Encryption, *Advance Security*, Oracle Database 11g, Oracle.
11. Transparent Data Encryption, Data sheet, OpenEdge Progress.
12. Cindy Papas, Chad Schieken, John Bastow, *Deploying SQL Server 2008 R2 Based on Payment Card Industry Data Security Standards (PCI DSS)*, White Paper, ParenteBeard LLC.
13. William Stallings, Cryptography and Network Security, *Principles and Practices*, LPE, Pearson Edition, Third Edition – (2003).
14. Silberschatz, Korth, Sudarshan, Database System Concepts, Mc Graw Hill, *International Edition*, Fourth Edition.
15. Miles E. Smid, Dennis K. Branstad, The Data Encryption Standard Past and Future. Available at https://pdfs.semanticscholar.org /3bb4/fdef3646b1170ab85aa155d287aac08079e2.pdf.