Journal of Computer Sciences and Applications, 2017, Vol. 5, No. 1, 17-24
Available online at http://pubs.sciepub.com/jcsa/5/1/3

DOI:10.12691/jcsa-5-1-3

Data Warehouse Signature: A Framework for Implementing Security Issues in Data Warehouses

Mayada J. AlMeghari*

Information Systems Department, Faculty of Computers & Information, Cairo University, Cairo, Egypt
*Corresponding author: mayada@teachers.org, malmeghari@ptcdb.edu.ps

Abstract Today, with the increase in the number of business web application users, the data size is obviously getting quiet huge. This data is composed together from integrated heterogeneous data sources to form Data Warehouse (DW). DW consists of critical and sensitive data used in many intelligent business models for decision support processes. Therefore, retaining data security for DW systems is a very important issue for researchers; also, it is the first priority for organizations. The current paper presents a new framework for implementing security issues in DWs named Data Warehouse Signature (DWS). The DWS framework ensures the security issues, which are Confidentiality, Integrity and Availability (CIA). In fact, this approach solves the security problems such as network overhead and data changing that occurs during transmission on Client-Server basis. The DWS approach achieves high performance by using parallel computing through a middleware named View Manager Layer (VML). Finally, this paper also presents the elastic architecture of the DWS framework with a vast applicability in multiple DW organizations depending on its network infrastructure.

Keywords: *security issues, data warehouse, data encryption, data integrity, data availability, hash function, middleware*

Cite This Article: Mayada J. AlMeghari, "Data Warehouse Signature: A Framework for Implementing Security Issues in Data Warehouses." *Journal of Computer Sciences and Applications*, vol. 5, no. 1 (2017): 17-24. doi: 10.12691/jcsa-5-1-3.

1. Introduction

The security issues are very important for business flow, especially, while using the financial data and business e-commerce models. The potential increase of internet users and multiple types of network technologies open ways for researchers to develop multiple types of middleware. This increase leads to the increase of data storages that are connected together for query of many business organizations. Data analysis is the main goal for decision-making process from the huge amount of data stored in DWs. To put this data in a security state in a web-based network, there is a need of a solution covering all the security issues in the DW systems. The security issues contain three attributes of data which are Confidentiality, Integrity and Availability called (CIA) [1,2].

To ensure data confidentiality, there are cryptography algorithms for data encryption/decryption, where the hacker is unable to reach the original data from the encrypted data. The data integrity uses mathematical methods of hash functions such as message digest or hash codes. When the hacker changes the data during transmission process, the hash function is computed in order to compare whether the data is changed or not. The data availability uses the middleware techniques as a cluster or grid in parallel computing to reach high performance. Thus, the Internet web system is available anywhere and anytime.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 presents the two models of the DWS framework: DWSend model and DWReceive model. Section 4 presents DWS key security generation. Section 5 presents the VML middleware of DWS approach. Section 6 presents the architecture of the DWS framework. Section 7 presents the DWS high performance and evaluation, Finally, Section 8 presents the main conclusion and future work.

2. Related Work

There are many approaches to study the security issues CIA and apply them in DW systems. Some of the approaches focused on ensuring data confidentiality and data integrity. In [3], a schematic technique using data type preserving encryption is proposed to ensure data confidentiality in DWs. This technique is applied in a multifaceted DW. The authors of [4] proposed a Fast Comparison Encryption (FCE) as a symmetric encryption algorithm applied in a traditional business DW. In [5], Deshmuk, A. and Qureshi, R proposed Transparent Data Encryption (TDE) technology applied in Microsoft SQL Server 2008. They used the encryption algorithm AES to ensure data confidentiality. Also they ensured data integrity by applying hash functions and digital certificates. In 2015, a multi-secret sharing approach is proposed for DWs in the cloud and allowing on-line analysis processing [6]. It is based on two schemes to ensure data

confidentiality by using homomorphic and incremental encryption algorithms. In this approach, data integrity is ensured with inner and outer signatures. It used the data environment of the SSB (Star Schema Benchmark) shared DW.

To solve network overheads, other DW security approaches used middleware to ensure data availability in the DW systems. The Specific Encryption Solution tailored for Data Warehouses (SES-DW) is proposed by Santos, R. J., et al. [7]. They used a cluster middleware for applying their algorithm in the Sales DW environment. The work in [8] presented a Modulus-Based Technique for Data Masking in DW (MOBAT). It used the MOBAT-SA middleware as a cluster type. This approach developed a black box, which has the keys generated for the masking data in DWs. In [9], the authors proposed Automatic Code Engendering (ACE) Algorithm. They used a cluster middleware applied in the mobile environment system by generating code of six length included numbers and alphabets. In 2015, Preeti and Khatka, K. [10] proposed Transposition-Substitution-Folding-Shifting Encryption (TSFS) Algorithm. The TSFS used a cloud middleware applied in WebOS virtual desktop.

Due to the increase of using information systems, many agencies from very diverse fields are connected with businesses. This has created many sets of separated databases or different data sources joined in a DW. Thus, the flow process of transmission between DWServer and Client takes a lot of time where the query result is a huge amount of data. This makes it difficult to transmit the data securely and fast. The data security problem appears in a long time of the encryption process for a huge amount of data in the DW besides its transmission time that causes overhead query response time [11].

Also, there is a need to apply data integrity as one of the security issues in DW systems. Integrity ensures that data is protected from malicious changes during sending by checking the hash code. This is an additional load to the encryption/decryption process, which leads to a crisis in the server-side. Thus, it cannot continue to provide services to users because of the overhead in the network. This paper solves two general security problems occurred through the transmission process between DWServer and Client: the network overhead problem such as query response timed out and the problem of changing data. In the current paper, a new approach named Data Warehouse Signature (DWS) is presented to solve the two security problems. The DWS is a framework for implementing and evaluating security issues in DWs.

3. Data Warehouse Signature Models

The DW stores a huge amount of data records which means large distributions on the internet. Therefore, the DWS framework is designed by using Client-Server architecture model. From the data flow between DWServer and client, there are two process models. These models are DWSend model and DWReceive model that use View Manager Layer (VML) middleware. The output of DWSend model includes the encrypted query result table and the encrypted final hash code for it. This output is an input for DWReceive model on the client-side. The client decrypts the received hash code and then compares it with the computed hash. The two models are described in the following two subsections:

3.1. DWSend Model

Based on the DW architecture, data is collected from many data sources at least three data sources. Then, this data is stored in one large data repository called DW. The Query Result Table (QRT) is a result of a SQL select statement that is executed by the client accessed from DWServer. Often, the QRT contains a large number of records, which may lead to two problems during sending the QRT to the client.

The first problem is the query response time overheads in DWServer. This problem is solved by using middleware named Server VML (SVML). The second problem is the process of ensuring that data is not changed and not allowed for a hacker to crack the data during transmission process. It is solved through computing the hash code function for encrypted data.

In this model, the DWServer collects the QRT from DW as a view table with a large number of records features. It separates the QRT into Block Objects (BOs) with a fixed size of number of records. These blocks are sent to SVML middleware to distribute into a set of executors.

By using parallel computing, the SVML middleware manages and executes two processes as shown in Figure 1. The first process is data encryption, where all the distributed BOs that has the same number of records are encrypted by using AES 128 bit algorithm [12]. This algorithm encrypts these blocks with the Share Key (SK) generated by using Diffie-Hellman (D-H) [13]. The second process is computing the hash code function, where the hash code of Encrypted Block Object (EBO) is computed by using SHA-1 algorithm [14]. The result of the first process is recollected according to the original content ordering to get the Encrypted Query Result Table (EQRT).

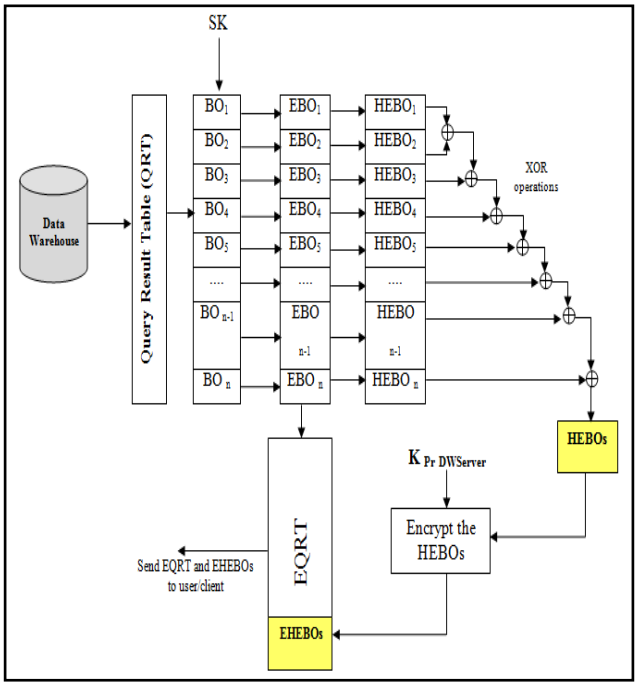

Figure 1. DWSend model – Data Encryption Process

Also, the results of the hash codes are recollected by the executive operations (XOR gate) to generate the final Hash of the EBOs (HEBOs). This hash value HEBOs is encrypted by using the RSA algorithm with the private key of DWServer to generate the Encrypted Hash of the EBOs (EHEBOs). Finally, the EQRT is concatenated with the EHEBOs to become one item sent to the client-side.

3.2. DWRecieve Model

This model occurs on client side by using Client VML (CVML) middleware to raise the burden of decryption from DWServer as shown in Figure 2. In this model, the EQRT is separated into EBOs of the same size in the DWSend model. The DWReceive model executes two processes in parallel computing through the CVML middleware. The first process computes the hash code of the EBOs by using a SHA-1 algorithm. The result of the hash codes is recollected by executive operations (XOR gate) to generate the final Hash of the EBOs (HEBOs). Then, this model compares between the two hash codes, the hash code of HEBOs sent from DWServer with the hash code of HEBOs computed in the Client side. If the two hash codes are the same, this means that the encrypted data of QRT is not changed when sending it from DWServer to Client. This model decides to apply the second process that decrypts each EBO by using AES algorithm with the SK key to get the Decrypted Block Object (DBO) [12]. Then, the BOs are recollected to get the QRT as ordering. If the result of the comparison is not the same, this model will not allow to decrypt the cipher blocks and stops displaying the QRT.

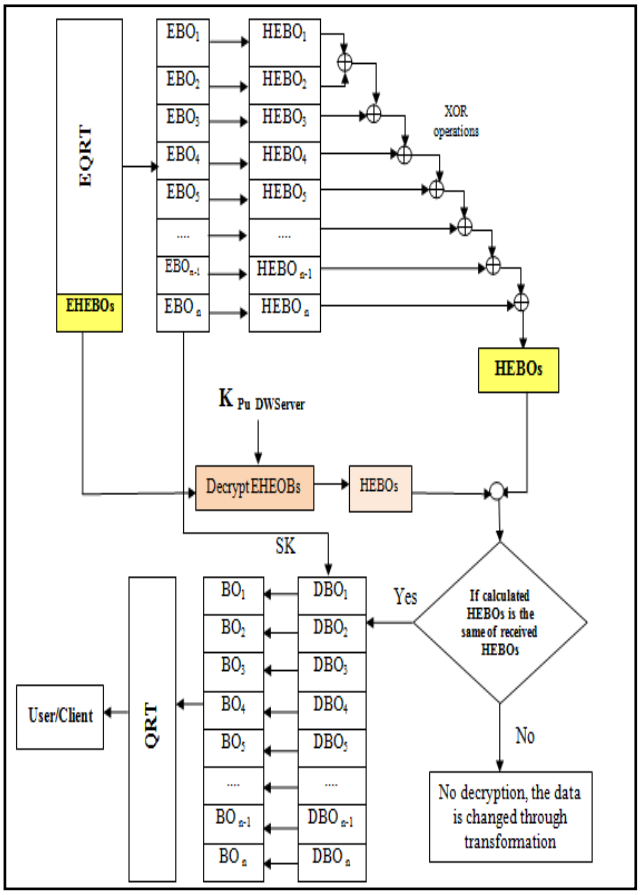

Figure 2. DWReceive model – Data Decryption Process

4. DWS Key Security Generation

The authentication and authorization in the DW systems are important challenges facing security issues. The authorization process allows the user to get access to data in the DW. This access is managed by generating the keys in the two DWS models. The session keys for DWServer and Client are the same keys used in symmetric encryption/decryption process of the AES algorithm. These keys are distributed by applying D-H algorithm. The authentication process allows the user to authenticate the source or destination of data (i.e., where data comes from or where data goes to). This is ensured by using RSA algorithm in asymmetric encryption/decryption for the Hash of the Encrypted Block Objects (HEBOs) with Private Key(K_{Pr}) / Public Key (K_{Pu}) of the DWServer [15,16].

5. VML Middleware

In this paper, the VML middleware of DWS is used in distributed systems as homogeneous systems. The distributed system covers two types of service providers of systems. The First type is a data source provider, such as Oracle 10g, which has a set of nodes in network integrated together to become the shared data [17]. In this century, the number of Internet users increased with the increase of bandwidth of data transmission rate. Thus, the data sharing as a source is available every time, and anywhere. The second type is the CPU power source, where it has a set of threads running in parallel computing (pipeline programming) to different clients over network [18]. According to the two previous types, the VML middleware uses the network resources to become a virtual supercomputer.

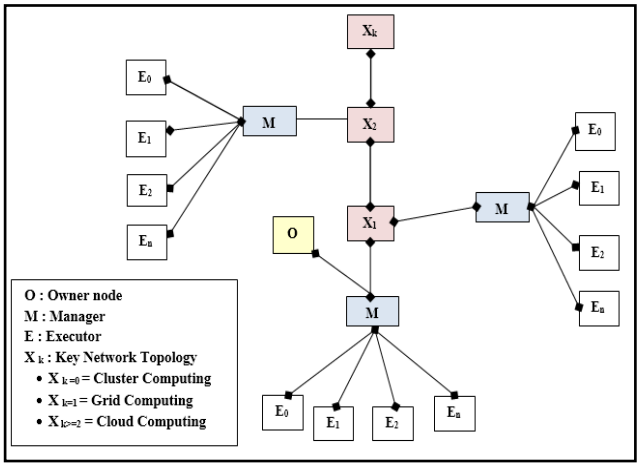

Figure 3. DW organization types for DWS VML middleware

There are many middleware used to execute the huge amount of data in parallel computing. The common large distributed middleware is called Alchemi. Alchemi is the open source middleware applied in many applications such as Tier Technologies, Global Data-Intensive Grid Collaboration, and CSIRO [19,20]. Depending on middleware network topology, the Alchemi middleware is simulated as VML middleware for DWS approach, and supported many types of DW organizations. If the DWS framework is applied in an organizational isolated

network, its middleware works in a cluster computing (private network organization). While, if the DWS approach is applied in distributed organizations, its middleware works in grid computing (public network organization). Otherwise, the DWS is applied in organizations Internet-based, its VML middleware works in cloud computing (Enterprise Desktop Network Organization), as shown in Figure 3.

The DW system consists of a number of operational databases joined together to provide a huge amount of data of QRT, which is requested from users. The DWS approach ensures data availability in DW security by using VML middleware. This middleware is applied on two sides, the DWServer side named SVML and the Client side named CVML. The VML middleware for DWS consists of three types of service scheduler process presented as follows:

(1) The Owner Job

In the DWSend model, the owner job is the DWServer that has a view of Query Result Table (QRT). It separates the QRT into blocks with the same size of the number of records. Also, this owner computes the current hash code value with the previous hash code value on the XOR gate to get the final HEBOs. The final hash is encrypted with the private key of DWServer.

In the DWReceive model, the owner job is the client that has the EQRT. It separates the EQRT into a number of blocks of the same size. Then, it computes the current hash code value with the previous hash code value on the XOR gate to get the final HEBOs. This owner decrypts the EHEBOs sent from the DWSend model by using the public key of DWServer. It compares the received hash code with its computed hash code. If the two hash values are the same, the data of the EBOs are decrypted to view to the client as QRT.

(2) The Manager Job

The manager is the main node in the middleware network. On the DWServer side, the manager is responsible for distributing the BOs to a group of executors. This manager returns the hash value for each EBO, and recollects all EBOs to become EQRT and then sent it to the DWServer. On the Client side, the manager returns the computed hash code value for each EBO, and recollects the decrypted BOs to become QRT, which is sent to the client.

(3) The Executor Job

On the two sides (DWServer and client), the executor is the worker that executes the encryption/decryption process on its BO. Then, this executor computes the hash code value for EBOs.

6. DWS Architecture

In this paper, the DWS approach ensures the security issues in DW systems through the two models (DWSend and DWReceive). The DWS has four components, which are integrated with the practical work in web business flow as shown in Figure 4. These components are namely below:

a) Client Application: The client/user on the internet web application sends a query request from DWServer. The client has a Session Key (SK) shared with the DWServer. The SK is generated by using a D-H algorithm. This key is used for encrypting the data of QRT by using symmetric algorithm AES. The client uses the public key of DWServer to decrypt EHEBOs. This means that the QRT is only coming from the DWServer.

b) DWServer: DWServer supports the collection of at least three operational databases. Thus, the QRT is a huge amount of sensitive data, so it needs a solution that ensures the security issues. The DWS approach uses a set of algorithms. The AES algorithm is used to encrypt/decrypt data of QRT for ensuring data confidentiality. The D-H algorithm is used to generate the share key for data encryption. The RSA algorithm is used to encrypt/decrypt the final hash code value. The SHA-1 algorithm is used to compute the hash code function for ensuring data integrity. The DWServer uses the SVML middleware, which distributes and schedules the execution of these algorithms in parallel to ensure data availability.

c) VML Middleware: It is a set of functions and operations concatenated together in distributed systems applied in network architecture. The data resources such as data sharing and the execution resources such as CPU power are the main elements used in distributed systems like middleware. In the middleware, there are three types of the distributed computing, depending on the network topology, such as cluster, grid, and cloud. The DWS framework has two models distributed in two sides, DWServer and Client. This paper presents two types of middleware such as SVML on the DWServer side and CVML on the client side to execute the encryption/decryption process and compute the hash code function.

d) Data Warehouse: The increasing number of organizations generates different data sources integrated together in a central data repository named DW. The data size in DW is larger than the data in operational databases. In addition, the number of users who use DW is bigger than the other data storages. The DWS approach discusses and implements the security issues to support the user of DW against attacking and cracking of data.

The data business flow in the DWS architecture is distributed between both the DWServer and the Client sides. As shown in Figure 4, there are a set of steps to represent the data flow in the DWS framework listed as follows:

1. The Client/User sends a query request to the DWServer.
2. The DWServer gets the data of QRT from DW.
3. In SVML middleware, the owner (DWServer) separates the QRT into BOs of the same size of the number of records and sends these blocks to the Manager.
4. The Manager distributes the BOs to a group of Executors.
5. The executor encrypts the data of its BO by using AES algorithm [12]. Also, it computes the hash value of its EBO by using SHA-1 algorithm [14]. Then, it sends the EBO and the hash code of EBO to the manager.

6. The manager sends the hash code of EBO and recollects the EBOs to become EQRT. Then, the manager sends it to the DWServer.
7. The DWServer computes the current hash value with the previous hash value using the XOR operation until getting the final hash value of the last EBO. Then, it encrypts the final hash HEBOs by using its private key to generate EHEBOs [16].
8. The DWServer sends the EQRT and EHEBOs to the client.
9. In the CVML middleware, the owner (Client) separates the EQRT from EHEBOs.
10. The Client decrypts EHEBOs by using the public key of DWServer to get the HEBOs as the Received Hash [16].
11. The Client divides the EQRT into EBOs and sends it to the Manager.
12. The manager distributes these EBOs into a group of executors.
13. The executor computes the hash value of its EBO to get HEBO and decrypts this block.
14. The executor sends HEBO and DBO to the manager.
15. The manager resends HEBOs to the Client and rearranges the blocks as one object to get the QRT.
16. The Client computes the current hash value of HEBO with the previous hash value of HEBO using the XOR operation to get the final hash value HEBOs as the Calculated Hash.
17. The Client compares between the Calculated Hash value and the Received Hash value. If the two hash values are the same, the manager resends the QRT to be viewed to the Client. Otherwise, an error message is displayed as data integrity error.

7. DWS High Performance Evaluation

Before enhancing DWS, in the current systems, the encryption process and the hash computing process are executed in serial, as shown in Figures (5-A, 6-A). This leads to overhead in the network and query response timeout. Some of DW security approaches used the execution processes in parallel such as SES-DW [7] and MOBAT-SA [8], where the encryption/decryption processes are executed in parallel computing. However, these approaches did not use the hash computing process to ensure data integrity as one of the security issues. Therefore, we need to enhance the DW systems by using the DWS models. The DWSend model has two main processes: data encryption and the hash computing. These processes are executed in parallel by using the SVML middleware. First, the QRT is divided into blocks of a number of records BOs. These blocks are distributed into a group of executors to be encrypted to ensure data confidentiality. Then, these executors compute the hash code of their EBOs to get the final hash code. This hash code is sent to the Client to ensure data integrity. Also, the DWS approach achieves high performance in average execution time as shown in Figure 5-B.

The DWReceive model has three main processes: the data decryption, the hash computing, and the hash comparing. This model divides the EQRT into blocks of the same size as EBOs. In CVML middleware, these blocks are distributed into a group of executors. These executors compute the hash code value of their EBOs and then decrypt these EBOs in parallel to reach high performance in average execution time, as shown in Figure 6-B. Finally, the DWReceive model compares between the two hash code values: the calculated hash and the received hash to ensure data integrity in the DWS system. If the comparison result is true boolean, the QRT is displayed to the client/user.

8. Conclusion and Future Work

In the digital age, the increase of internet users and the speed of data transmission rate have become the main factors for the availability of the web security system. In the distributed systems, there are methods to use data sources and CPUs that are distributed in the network as middleware to virtual supercomputer. This paper has proposed a new approach called DWS as a framework for implementing security issues in DWs.

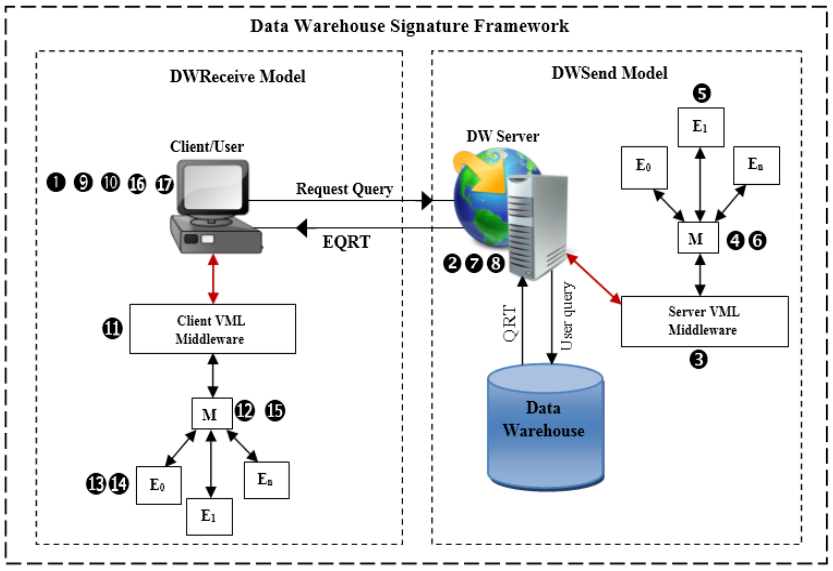

Figure 4. DWS Architecture and its data flow

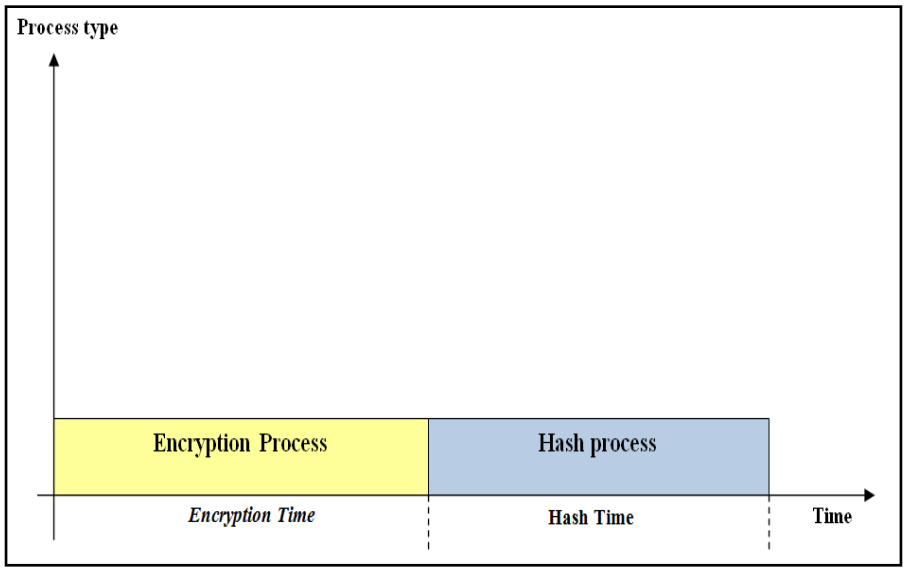

Figure 5A. Total send time in the serial current systems

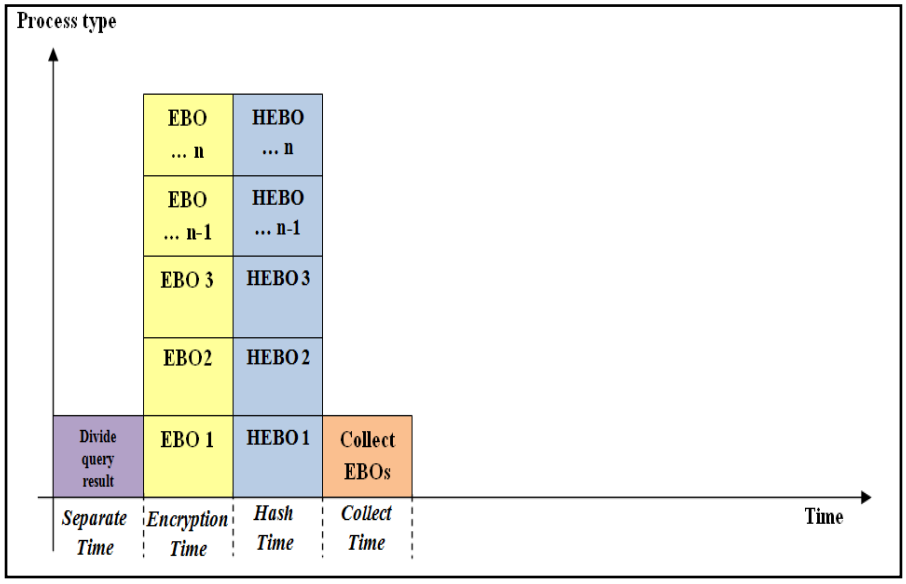

Figure 5B. Total execution time for DWSend model in DWS framework

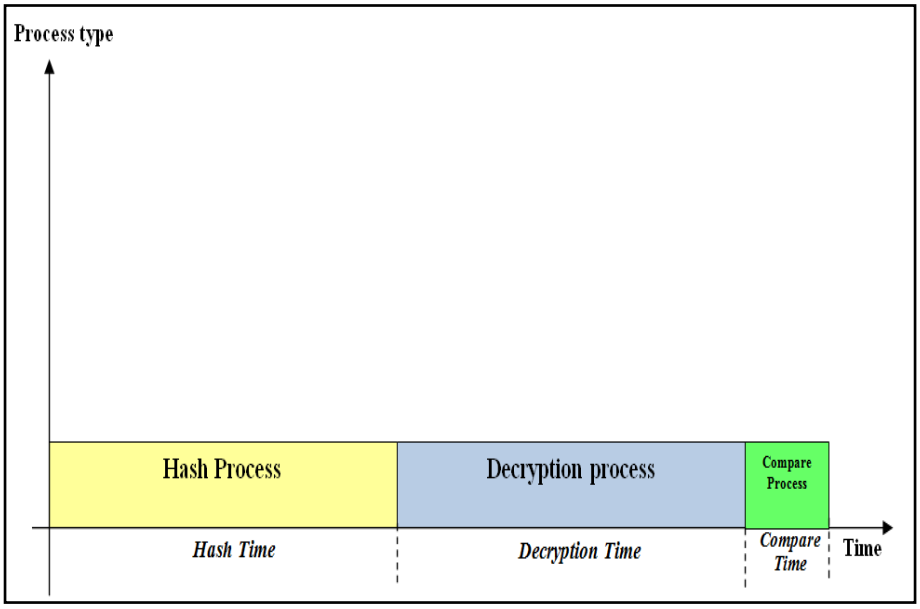

Figure 6A. Total receive time in the serial current systems

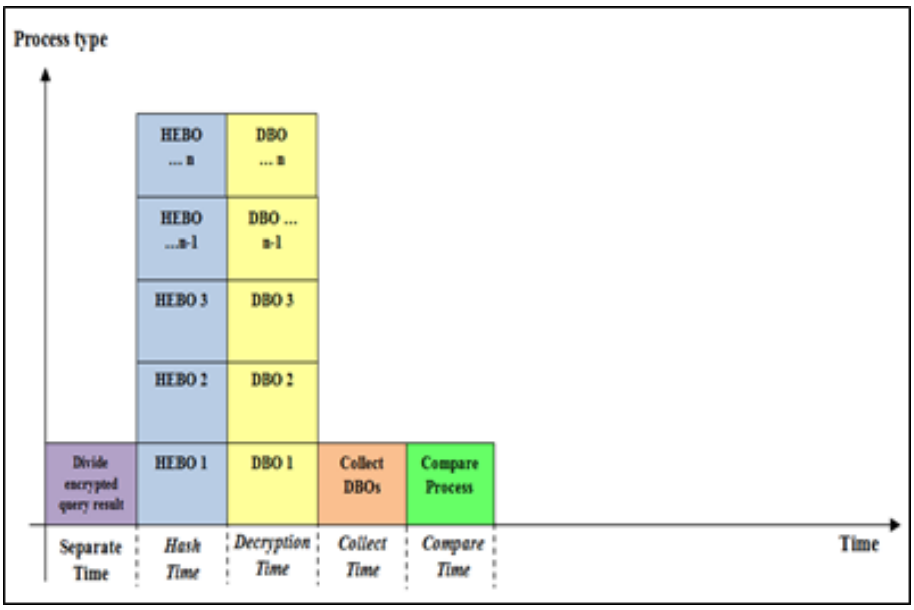

Figure 6B. Total execution time for DWReceive model in DWS framework

This approach ensures all the three security issues, which are Confidentiality, Integrity and Availability of data in DW systems. The DWS framework that has applied two models, DWSend and DWReceive, has achieved high performance in parallel execution time. The execution of encryption/decryption process and hash computing process of data as blocks in parallel saves time more than the execution of the two processes in serial. In future work, the DWS approach needs to detect the basic high performance evaluation factors. The first factor is to find the query memory buffer for the VML middleware. While the other one is to evaluate the high performance when the number of executors increases in the VML middleware of the DWS approach. Finally, we will analyze the DWS approach in different threat models to develop the authentication property and key distribution for more confidentiality in trusted DW systems.

Acknowledgements

I would like to thank Allah, my family, my friends, my supervisors, and my colleagues for their support, motivation, and help during this research.

References

[1] Aleem, S., Capretz, L. F., and Ahmed, F., "*Security Issues in Data Warehouse*," Recent Advances in Information Technology, pp. 15-20, Canada 2015.

[2] Konda, S. and More, R., "*Augmenting Data Warehouse Security Techniques - A Selective Survey*," International Research Journal of Engineering and Technology (IRJET), Volume 02, Issue 03, pp. 2209- 2213, June 2015.

[3] Reddy, M.S., Reddy, M.R., Viswanath, R., Chalam, G.V., Laxmi, R. and Rizwan, Md.A., "*A Schematic Technique Using Data type Preserving Encryption to Boost Data Warehouse Security*," International Journal of Computer Science Issues (IJCSI), Vol. 8, Issue 1, pp. 460-465, January 2011.

[4] Ge, T. and Zdonik, S., "*Fast, Secure Encryption for Indexing in a Column-Oriented DBMS*," Proceedings of the 23rd International Conference on Data Engineering (ICDE), Istanbul, pp. 676-685 IEEE, May 2007.

[5] Deshmuk, A. and Qureshi, R., "Transparent *Data Encryption-Solution for Security of Database Contents*," International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 2, No.3, pp. 25-28, March 2011.

[6] Attasena, V., Harbi, N., and Darmont, J., "*A novel multi-secret sharing approach for secure data warehousing and on-line analysis processing in the cloud*," International Journal of Data Warehousing and Mining (IJDWM), Vol. 11, No. 2, pp. 22-43, 2015.

[7] Santos, R. J., Rasteiro, D., Bernardino, J. and Vieira, M., "*A Specific Encryption Solution for Data Warehouses*," Proceedings of International Conference on Database Systems for Advanced Applications. Springer Berlin Heidelberg, pp. 84-98, April 2013.

[8] Santos, R. J., Bernardino, J., and Vieira, M., "*A data masking Technique for Data Warehouses*," Proceedings of the 15th International Database Engineering and Applications Symposium (IDEAS'11), ACM, pp. 61-69, 21-27 September, 2011.

[9] Chowdhury, R., Chatterjee, P., Mitra, P. and Roy, O., "*Design and Implementation of Security Mechanism for Data Warehouse Performance Enhancement Using Two Tier User Authentication Techniques*," International Journal of Innovative Research in Science, Engineering and Technology, Volume3, Special Issue 6, pp. 165-172, February 2014.

[10] Preeti and Khatka, K., "*Enchancing Data Security And Privacy On WebOS Using TSFS*," International Journal of Computer Engineering And Electronics Technology (IJCEET), Vol. 1, Issue 1, pp. 1-5, 2015.

[11] Santos, R.J., Bernardino, J. and Vieira, M., "*Balancing Security and Performance for Enhancing Data Privacy in Data Warehouses*," Proceedings of the 10th International Conference on Trust, Security and Privacy in Computing and Communications, Changsha, DOI 10.1109/TrustCom.2011.33, pp. 242-249, IEEE, 16-18 November 2011.

[12] Federal Information Processing Standards Publication FIPS 197, "*Advanced Encryption Standard (AES)*," pp. 13-26, November 2001.

[13] Stallings, W., "*Cryptography and network Security Principles and Practice*," Sixth Edition, Pearson Education, Inc, ISBN 13: 978-0-13-335469-0, pp. 287-291, 2014.

[14] Federal Information Processing Standards Publication FIPS 180-4 , "*Secure Hash Standard (SHS)* ," pp.3-20, March 2012.

[15] Bhanot R, Hans, R., "*A Review and Comparative Analysis of Various Encryption Algorithms*," International Journal of Security and Its Applications Vol. 9, No. 4, pp. 289-306, 2015.

[16] Kak, A., "*Lecture 12: Public-Key Cryptography and the RSA Algorithm*," Lecture Notes on "Computer and Network Security, Purdue University, pp. 3-7, April 2015.

[17] Oracle Corporation, "Oracle *Database Administrator's Guide-10g Release 2 (10.2)*," B14231-02, p. 155, May 2006.

[18] Suleman, M.A., Qureshi, M.K., Khubaib and Patt, Y. N., "*Feedback-Directed Pipeline Parallelism*," Proceedings of the 19th International Conference on Parallel Architectures and Compilation Techniques (PACT'10), Vienna, Austria, Copyright ACM 978-1-4503-0178-7/10/09, pp. 147-156, September 2010.

[19] Bhatnagar, R. and Patel, J., "*An Empirical Study of Security Issues in Grid Middleware*," International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 4, Issue 1, pp. 470-474, January 2014.

[20] Luther, A., Buyya, R., Ranjan, R. and Venugopal, S., "*Alchemi: A .NET-based Enterprise Grid Computing System*," Internet Computing, cloudbus.cis.unimelb.edu.au, pp. 5-9, 2005.