ISSN:0975-9646

Analysis of Security Techniques Applied in Database Outsourcing

Ajeet Ram Pathak[1], B. Padmavathi[2]

[1,2]*Department of Computer Engineering,*
University of Pune
[1,2]*G. H. Raisoni College of Engineering and Management,*
Pune, India

***Abstract*—Database Outsourcing is a nascent data management paradigm in which the data owner stores the confidential data at the third party service provider's site. The service provider is responsible for managing and administering the database and allows the data owner and clients to create, update, delete and access the database. There are chances of hampering the security of the data due to untrustworthiness of service provider. So, to secure the data which is outsourced to third party is a great challenge. The major requirements for achieving security in outsourced databases are confidentiality, privacy, integrity, freshness in case of dynamic updates, access control in multi-user environment, availability and query authentication and assurance. To achieve these requirements various security mechanisms like encryption based approach, fake tuple based approach, customized secret sharing approach, authenticated data structure approach, aggregate-able signature schemes have been put forth till date. In this paper, various mechanisms for implementing security in outsourced databases are analyzed along with their usefulness in a great detail.**

***Keywords*— Access Control, Confidentiality, Freshness, Integrity, Outsourced Databases, Query Authentication,**

I. INTRODUCTION

The large amounts of data are generated in the small-scale and large scale sectors of companies, government sectors, education and scientific institutions. It becomes very important for these organizations to maintain and administer those data. That's why the companies adopts option of purchasing and installing the data management tools and manages their own data. This in-house method of database management suffers from various problems. The company has to hire the database professionals, upgrade the system time to time and keep the data up-to-date. Also during version change, the system's availability gets hampered. Due to these reasons, companies have to focus on data management rather than their business logic. This leads to huge amount of expense on data management related activities, deployment and operation.

To ameliorate the problem of data management for in-house databases, the new approach came into picture, namely, outsourcing database to the third party provider. Due to advancement in cloud storage technology, network technology and improvements techniques in data transmission techniques, database outsourcing has attracted the attention of organizations across the globe. The term "Database as a Service" (DBaaS) appeared in [1]. DBaaS is the breakaway technology of the recent era. Cloudant, Amazon DynamoDB, Hosted MongoDB are some examples of database service providers. DBaaS is a database management concept in which the data owner of the organization stores their data at the third party service provider's site and delegates the responsibility of administering and managing the data to the service provider. This paradigm alleviates the need of installing data management software and hardware, hiring administrative and data management crew (personnel) at the organization's site. Due to this, the organization can concentrate on their core business logic rather than on the tedious job of data management leading to the saving in data management cost.

As the data is stored at the service provider's site, it may be the case that service provider is distrustful in terms of disclosing and misusing the data. In this case, security of the database can be hampered in a dramatic way. If proper security is not enforced, then there are chances of data breaches and hacking the data in an unauthorized manner. Data breaching means disclosing the sensitive data intentionally or unintentionally. According to the survey taken by Trustwave Global Security[1], out of 450 data breach samples, 63% of investigations were related to the administration of third party service providers. Therefore, to enforce the security perspective in outsourced databases is the need of all companies which are outsourcing their data to service provider.

Once the database is outsourced, security services must be availed to the clients who use the database service. Security service is referred to as the capability which supports various security requirements like CIA triad (*Confidentiality*, *Integrity* and *Availability*), *correctness*, *freshness*, query authentication and assurance. Security services are considered as the superset of AAA (*Authentication, Authorization and Accounting*). They are implemented by using security mechanisms. Security mechanisms include encryption, digital and aggregate-able signature schemes, customized secret sharing schemes, etc. This paper analyzes such security mechanisms which are applied in outsourced databases.

The contents of the paper are portrayed as: The section II deals with the brief introduction of DBaaS and the architecture of outsourced database model. The key requirements of security in outsourcing are given in section III. The security techniques provided for data outsourcing along with their pros and cons are analyzed in section IV. The paper is concluded in section V along with the possible enhancements possible.

[1]http://www.computerweekly.com/news/2240178104/Bad-outsourcing-decisions-cause-63-of-data-breaches

II. BACKGROUND

This section describes the concept of DBaaS and benefits, architecture of database outsourcing model, challenges associated with the same.

A. Database as a Service

According to the technopedia[2], DBaaS can be defined as

"A Cloud computing service model that provides users with some form of access to a database without the need for setting up physical hardware, installing software or configuring for performance".

All of the administrative tasks and maintenance are taken care of by the service provider so that all the user or data owner needs to do is exploit the efficient and agile database functionality provided as a service. The Organizations pay for the database service they are getting from the service provider. For the companies with less amount of resources limited hardware and time-bound projects, DBaaS best suits the scenario. Due to its inherent scalable property, DBaaS can scale up well in case of increasing user demands and also scale down when the demand subsides. The deployment of infrastructure for industries gets easier with the help of DBaaS. It offers flexible and on-demand services, optimizes performance tuning of the system, lowers the operating cost and complexity, accelerates the provisioning i.e. allows to clone the old database with a new schema, shortens the sales cycle, provides failover environment for project execution, enables the centralized administration and management of all kinds of databases. DBaaS incorporates the Quality of Service by removing the data and database redundancy. Considering the adoption of DBaaS in industries[3], the research states that in 2016, the revenue generated by DBaaS providers will be $1.8 billion which is almost twice of the revenue generated in 2012 which is $150 million as shown in figure 1.

B. Architecture of Outsourced Database Model

Fig. 2 depicts the architecture of Outsourced Database Model. It consists of 3 main entities as

- Data Owner
- Service Provider
- Clients

Generally, data owner and clients are considered as trustful entity while service provider is distrustful in context of disclosing data in an unauthorized manner. A data owner uploads their organization data at third party service provider's site using high speed communication link. A data owner can insert new data, modify the existing data and delete data. In case of multiple clients, he can set access level permissions for using the data. The service provider stores the data uploaded by the data owner. Data management hardware and software tools are deployed and maintained at the provider's site. To ensure the availability of data in case of data crash or version change, standby servers are also maintained due to which seamless and uninterrupted database service is provided.

[2]http://www.techopedia.com/definition/29431/database-as-a-service-dbaas
[3]https://451research.com/report-short?entityId=78105&referrer=marketing

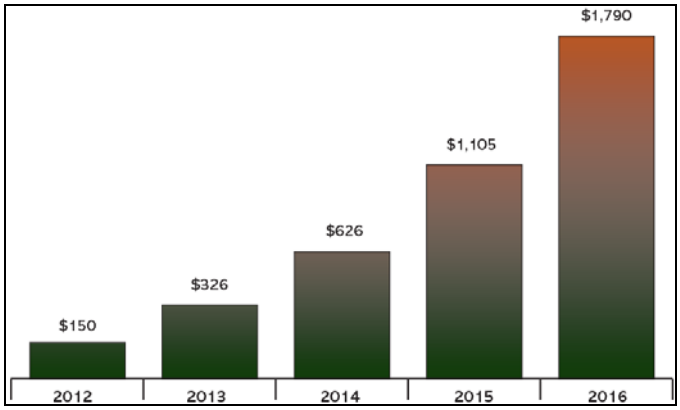

Fig. 1. DBaaS Market Revenue and Forecast ($ Million) [3]

To ensure the availability of data in case of data crash or version change, standby servers are also maintained due to which seamless and uninterrupted database service is provided. When client or data owner fires the query on database, the query is processed efficiently and results are sent back to the querier. The clients are given permission to access the data according to their privilege level.

There are 3 types of outsourced database model which are categorized on the basis of number of data owners and clients involved. The first model is *unified client model* in which the database is used by single entity i.e. here functionality of client and data owner is same. The data owner does all the operations on the database. The communication link between data owner and client has high bandwidth. This model is adopted in [2], [3], [4]. The second type of outsourced database model is *multiple client model* where multiple clients are given the authority of read only access. Here, the database can be accessed through mobile devices, laptops, PCs with limited bandwidth communication link. This model is adopted by [2], [3], [5]. *Multiple data owner model* is the third type of model which is adopted in [6]. In this model, each data owner uploads data at service provider's site. So, for every group of data owner and client, the separate access control and security policies are needed to be applied. This model can also be called as *multi-authority outsourced database model*.

III. KEY REQUIREMENTS OF SECURITY ASPECTS IN OUTSOURCING

We have discussed the key requirements of security aspects in database outsourcing in this section.

- *Confidentiality* assures that only the authorized and intended users or systems are given consent to access the data. *Data confidentiality* refers to keep the data concealed from unauthorized access when it is stored and also when the data is in transit state. While ensuring the *confidentiality*, some additional dimensions are considered viz. *user privacy* and a*ccess privacy*. Privacy is one of the primary requirements to achieve the security. User *privacy* conceals the user identity when he fetches or manipulates the data. Access *privacy* is assured when the access pattern of database and intended database records for a particular user are kept secret.

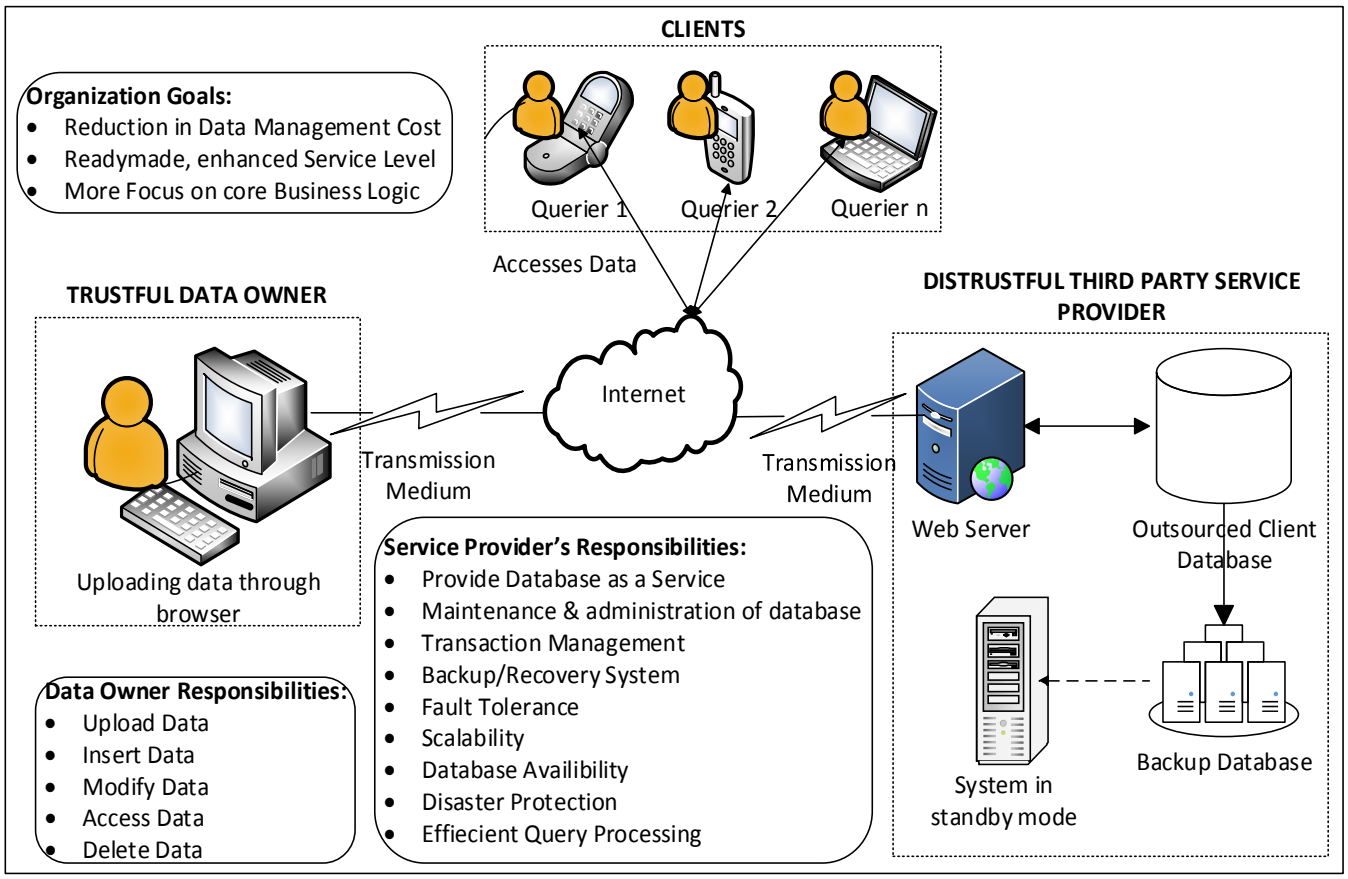

Fig. 2. Architecture of Outsourced Database Model

- *Integrity* assures that the data stored in database or in transmission state are not modified or manipulated except by trusted persons or processes. *Completeness* and *correctness* are two important dimensions of *integrity*. *Completeness* means that query results obtained by fetching all records from the database and no any record containing the predicate in the query is excluded. It ensures that entire results are obtained when a query is fired on the database. *Correctness* means that the query results obtained from server are tamperproof and generated by original server. To verify that *integrity* is maintained, *query assurance* mechanism needs to be incorporated which ensures that query fired against the database is correctly and completely executed by service provider or process including all matched predicates in query.
- *Availability* ensures that data are available to the trusted users and systems when they access database in an authorized manner. It is degree or extent to which database is in operable state which is calculated in terms of reliability. It is recommended for service providers to provide always-on availability of database to their valued and authorized users. To provide high level of availability, database system's down-time should be kept low.
- *Authenticity* implies that contracts, query transactions and communication are genuine and identities of the involved entities (users and system) are verified and known. To provide the *authenticity*, the digital signatures are used.
- *Freshness* of query results forms an important aspect of security when the database is periodically updated by the data owner. In such dynamically changing databases, *freshness* guarantees that the query results are obtained by executing the queries over the most updated database.
- A person who works with data is delegated some responsibilities of data assurance. The tasks for which that person is responsible are the part of data security plan. This is called as *accountability*. It assures that all the operations performed by the users, processes or systems can be traced and identified to the respective entity. This ensures that all the valuable assets are refrained from the illegal access. This is also called as *access control*. Access control can be implemented by assigning role to the individual so that they can access the data according to their privilege level only. Access control matrix, access control list and access control capabilities list are some mechanisms for achieving the same.
- *Risk management* is considered for proper implementation of security in outsourced databases. It includes the set of activities to identify and track the data security vulnerabilities and also the control measures are set to avoid the further risk to security. These terms are maintained in the security policy agreed by both the data owner and service provider to meet the data owner's organization needs.

These are the security requirements to be achieved in outsourced databases.

IV. Analysis of Security Techniques Applied In Outsourced Databases

In this section, all the security techniques in outsourced databases along with their pros and cons are explained.

A. Hardware-level Encryption based approach

To ensure the *data privacy*, special encryption hardware for IBM DB2 has been used in [1]. All the rows of the database are encrypted as a whole using DES (Data Encryption Standard) algorithm. The query execution time in hardware-level encryption is much less as compared to the software-level encryption. TrustedDB [7] is a server side hosted and robust prototyping hardware provides *privacy* and data *confidentiality* performing the query optimization and supports any type of query fired against database. But, due to in-built hardware processing of query, it suffers from cost overhead and it has some performance limitations.

B. Order Preserving Encryption based approach

To keep the secrecy of data, encryption is applied. But this encrypted data can't be easily queried. Order Preserving Encryption (OPE) provides a better way to maintain the order of the encrypted tuples (cipher text). The OPE approach in [8], [9] is as follows. Suppose E is an OPE function and P_1, P_2 are plain text values then $C_1 = E(P_1)$ and $C_2 = E(P_2)$ provided that if $P_1 < P_2$ then $C_1 < C_2$. These approaches suffer from plain-text chosen attacks. An index based programmable OPE approach is put forth in [10]. In this approach, input tuple x is mapped to the linear expression as $ax + b + noise$. The value of *noise* is selected such that order of input values will be retained. In this approach, input values are indexed by more than one linear expression making the approach brute-force attack tolerant. As no correspondence is maintained between distribution of indexes and distribution of input values, determining the range of input tuples is harder. This approach supports only the range queries.

C. Authenticated data structure based approach

To provide the *integrity* and authenticity for outsourced databases, authentication data structure based approach is used. These include implementing MACs (Message Authentication Code), Digital Signature scheme, MHT (Merkle Hash Tree).

MAC function takes secret key and a variable length data (message) as an input and produces the MAC code. It assures *integrity* and *authenticity* in *unified outsourced database model* where data owner and client are one and the same entities and where bandwidth overhead and computation overhead are also less. MAC is not suitable in *multiple clients* and *multiple data owner model*. In case of deceitful client with secret MAC key, he can collude with the server and add the fraudulent records to the database. So *non-repudiation* will not be achieved with MAC.

Digital signatures are used to provide *authenticity*, *integrity* and *non-repudiation* and it is used when a large number of entities are involved in outsourcing [11]. For implementing the digital signature scheme, large storage and bandwidth is required.

MHT is used for assuring the *integrity*, *secure verification*, *authentication* of larger data. It works as follows: Suppose data value is represented as $d_1, d_2,\ldots,d_n$. A leaf node N_i $(i = 1, 2,..n)$ stores the hashed value of data such that $N_1 = h(d_1)$ and so on. The non-leaf node is represented by concatenation of hash values of its children. MHTs are suitable only for range queries as the server returns only two full paths (Minimum value path and maximum value path) to the client.

The combination of MHT and B-tree is implemented by [5] known as Merkle B-tree to provide *integrity* (*completeness*, *correctness*). In this, client computes all the hashes of the sub-tree using *verification object* in a repeated manner till he reaches to the root of tree. Once he computes the hash of root, he can verify the *correctness* using owner's public key and hashed value of root. As the client is forced to find the hash of whole sub tree, it provides *completeness* assurance. A dithered B-tree which is combination of original B-tree and its corresponding dithered B-tree implemented in [12] based on key-value pairs. It prevents a third party from learning whether key is present in database or not, thus providing *privacy*. As the server stores two kinds of trees, it requires more space. Also communication cost for transferring the page-level data is also more.

Nested Merkle B^+ Tree which is an index structure is implemented by [13] to provide query assurance for dynamic outsourced XML databases. In this, all paths in XML document are listed out by tree traveler algorithm. The result contains leaf entries pointing to records, data records and not-in-result (co-path) entries. When the client fires query, server returns the root with signature and timestamp. Actual result and co-path are assembled in *verification object (VO)*. The client recalculates hast of root and verify it with signature for assuring *completeness* and *correctness*. The timestamp value of root is also returned by server. Client compares this timestamp with the timestamp published by data owner. If both the timestamp matches, then *freshness* verification is successful.

To ensure *freshness*, authenticated data structure approach is given in [4]. when client updates data at timestamp 't', he generates new certificate with same timestamp and put it in outsourced database. It also supports updating by multiple clients. In case of multiple clients, when a client certifies a root signature, it also provides a reference to the updated signature from another client.

D. Secret Share Distribution based approach

Though the encryption makes the data unintelligible for protection, it imposes extra overhead of encryption and decryption on the system and degrades the performance of database. So to protect the data, secret share distribution based approach best suits in the system where encryption is not applied. Rather than performing encryption on data, data is distributed on multiple servers, called as shares.

Shamir's Secret Sharing Scheme [14] is the basis of distribution based algorithms. This scheme is explained as follows. In a *k-out-of-n* secret sharing scheme, the secret value is divided into *n* parts and distributed among n entities. To reconstruct this secret, at least *k* entities are

needed to reveal the secret. Therefore it is also called as *(k, n)* threshold secret sharing scheme. It works on polynomial interpolation scheme. The data owner selects a *(k-1)* degree polynomial *P* such that $P(0) = S$ where *S* is the secret value to be shared. The polynomial is given as $P(x) = a_{k-1}x^{k-1} + \ldots + a_1x + a_0$ where $a_0=S$. If *k* entities altogether share the knowledge, they get k separate points on the curve and thereby obtain the secret share $S = P(0)$. If *k-1*entites share their knowledge, they can't retrieve the secret.

The advantage of using secret sharing scheme is that a single secret share holder can't change the data so single point vulnerability is removed achieving the tight control over the system. A customized secret sharing approach is used in [15], [16], [17], [18]. The theoretically secure and fault-tolerant *privacy* preserving approach using secret sharing is proposed in [15] in which exact match, range, aggregation and join queries are supported. In case of range queries, approach is prone to known database attack. This approach doesn't provide collusion-resistance facility. Confidentiality is not assured in case where a client with the secret value (Data) colludes with *k* data servers.

The dispersion of data among the data servers and use of interpolation in [16] achieved high level of security for *confidentiality* and *privacy*. This approach works well even for known database and frequency attacks. This approach has more computational overhead at the client side as compared to approach in [15]. The *completeness*, *correctness* and *freshness* are assured in [17]. In this, the data to be secured is stored on *n* data servers and references to those are stored on Index servers. By secret sharing scheme, *confidentiality* is achieved as the input value in terms of n-ary vector is only known to the trusted clients. At index server side, *confidentiality* is achieved by creating the B^+ tree over the searchable attributes and storing them in encrypted buckets. For *freshness* verification, aggregate signature scheme namely condensed RSA is used. The clients are considered as the trusted entities and so they know expected number of shares through encrypted buckets and hence *completeness* is verified. Using the extra-interpolation, *collusion resistance* is achieved. The drawback of this approach is that it only supports the numeric data. It does not support aggregate queries.

E. Attribute based access control approach

For multi-owner system *privacy* enhancing model is proposed in [6] providing accountability, authentication and authorization for all the access activities. In this, multi-agent system model is used which provides access rules for the users having the right to read and write data files.

For providing the access control in the cloud systems Ciphertext-Policy Attribute-based Encryption (CP-ABE) is applied in [19], [20], [21].These approaches does not require the distribution of keys by the data owner and allows the data owner to direct control the access policies. The approaches in [6], [19], [20] works only for the single authority systems and does not work for the multi-authority systems where a single person is accountable for more than one database authorities. Provable secure access control in multi-authority system is put forth in [21]. This method uses the token based decryption approach and provides forward and backward security. The advantage of this method is that computation and communication cost incurred for revocation is less. It suffers from one weakness. The attributes associated with the users are placed in Attribute Authority. The revoked user can corrupt this authority by updating their own secret key also the secret key of non-revoked users.

An efficient and fine-grained access control mechanism based on column-level access control is given in [22]. This mechanism alleviates the need of database re-keying by using token based key derivation in case of dynamic updates. In this, data owner gives the sub keys to the user to access that data column only which user access permission. The number of keys held by user in the system should be reduced to achieve the better performance.

F. Fake tuple based approach

The insertion of fake tuple based approach is adopted in [3], [23] and [24] to provide the *integrity* services. It mainly includes two approaches as probabilistic approach and deterministic approach.

In probabilistic method [3], the fake tuples are created and inserted into the database. For verifying the query *integrity*, the query is fired against the database server which contains both the real and fake tuples as the predicates. The server returns the query results. These results are verified by the client who knows all the fake tuples in the database. The client evaluates the fake tuples returned by server through result and the tuples determined by him. If tuples from server and from client are found out to be different, then the server is considered as dishonest and it is declared that the data has been tampered; else if tuples from both client and server are same, then it can be claimed that *completeness* is achieved i.e. *integrity* of the data is maintained. As already mentioned, the client should be aware of the fake tuples. The client has to maintain the copy of recent tuples. In case of large databases, a local database of fake tuples has to be maintained which causes extra storage overhead on client and it is against the concept of outsourcing. Freshness is guaranteed by using the fake update operation. The client deletes and inserts the fake tuples and analyse the results obtained by the server and evaluates the *freshness*. The deterministic approach [23] alleviates the need to save the fake tuples. The deterministic functions are used to recreate the fake tuples. These created fake tuples have prominent pattern and it can be easily noted by the hacker. Therefore, an encryption is applied on the real and fake tuples. Due to this, computation overhead increases.

The approach in [24] creates the tuples with no distinguishable pattern using uniform distribution and hence removes the need of encrypting the tuples. This approach does not provide *correctness* guarantees to the user.

G. Combined fragmentation and encryption based approach

To achieve the *confidentiality* and *privacy*, the combined approach of fragmentation and encryption is adopted in [25] and [26]. In this, data to be protected is split up into fragments and encrypted. These fragments are stored at same or different servers. The encryption key and location of servers is known to the authorized users only.

To limit the fragmentation level and to maximize the affinity between attributes of data, heuristic algorithms are used. The sensitive information and its associations are expressed in *confidentiality* constrained and modelled using heuristic approach. When the information stored in the fragments over time, it becomes overhead to protect and change the data in the fragments.

V. Conclusion and Future Directions

Database outsourcing is a nascent data management technology in the recent era which is accepted in the industries due to its inherent profitable features. In this paper, we have discussed the concept of DBaaS, its architecture and its benefits. We have mainly focused on how the security applied in outsourced databases and analyzed the techniques with their usefulness for the same.

The emphasis can be given on implementing security perspective in the domain of NoSQL databases. The future work can also be focused on providing security for outsourced database along with reducing the communication, computation cost. There is much scope for improving the optimization of query processing time. The generic system can be developed which efficiently provides DBaaS and works on any database providing all the security mechanisms.

References

[1] H. Hacigumus, B. Iyer and S. Mehrotra, “Providing database as a service,” in Proc. of IEEE 18th ICDE, 2002, pp. 29-38.

[2] E. Mykletun, M. Narasimha, and G. Tsudik, “Authentication and *integrity* in outsourced databases,” In Proc. of ACM Trans. on Storage, vol. 2, 2006, pp. 107-138.

[3] M. Xie, H. Wang, J. Yin, and X. Meng, “Integrity auditing of outsourced data,” VLDB 2007, pp. 782-793.

[4] Zheng-Fei Wang, Ai-Guo Tang, “Implementation of Encrypted Data for Outsourced Database”, In Proc. of Second International Conference on Computational Intelligence and Natural Computing (CINC), IEEE, 2010, pp. 150-153.

[5] Li Feifei, Marios H, George K, “Dynamic Authenticated Index Structures for Outsourced Database” In Proc. of ACM SIGMOD’06. Chicago, Illinois, 2006, pp. 121-132

[6] Somchart Fugkeaw, “Achieving Privacy and Security in Multi-Owner Data Outsourcing”, In Proc. of IEEE Transactions 2012, pp. 239-244.

[7] Sumeet Bajaj, Radu Sion, “TrustedDB: A Trusted Hardware based Database with Privacy and Data Confidentiality”, In Proc. of IEEE Transactions on Knowledge and Data Engineering, 2013, in Press

[8] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order pre-serving encryption for numeric data. In Proceedings of the 2004 ACM SIGMOD international conference on Manage-ment of data, SIGMOD ’04, pages 563–574, 2004.

[9] A. Boldyreva, N. Chenette, Y. Lee, and A. O’Neill. Order-preserving symmetric encryption. In Proceedings of the 28th Annual International Conference on Advances in Cryptolog y, EUROCRYPT ’09, pages 224–241, 2009.

[10] Dongxi Liu, Shenlu Wang, “Programmable Order-Preserving Secure Index for Encrypted Database Query” In Proceedings of 2012 IEEE Fifth International Conference on Cloud Computing, pp. 502-509, 2012.

[11] R. J. Morteza Noferesti, Mohammad Ali Hadavi, "A Signature-based Approach of Correctness Assurance in Data Outsourcing Scenarios," ICISS 2011, India, pp. 374-378.

[12] Chung-Min Chen, Andrzej Cichocki, Allen McIntosh, Euthimios Panagos, “Privacy-Protecting Index for Outsourced Databases”, In Proc. of ICDE Workshops 2013, pp. 83-87.

[13] Viet Hung Nguyen, Tran Khanh Dang, Nguyen Thanh Son , Josef Küng, “Query Assurance Verification for Dynamic Outsourced XML Databases”, Second International Conference on Availability, Reliability and Security (ARES'07), 2007, pp. 689- 696.

[14] A. Shamir. “How to share a secret” In Communications of the ACM, 1979, pp. 612–613.

[15] D. Agrawal, A. E. Abbadi, F. Emekci, and A. Metwally, "Database Management as a Service: Challenges and Opportunities," ICDE 2009, pp. 1709-1716.

[16] M. N. Mohammad Ali Hadavi, Rasool Jalili, "Secure Data Outsourcing Based on Threshold Secret Sharing: Towards a More Practical Solution," PhD Workshop at VLDB 2010, Singapore, 2010, pp. 54-59.

[17] M. A. Hadavi, M. Noferesti, R. Jalili, E. Damiani “Database as a Service: Towards a Unified Solution for Security Requirements” In Proceedings of IEEE 36th International Conference on Computer Software and Applications Workshops, pp. 415-420, 2012.

[18] Brian Thompson, Stuart Haber, William G. Horne, Tomas Sander, Danfeng Yao, “Privacy-Preserving Computation and Verification of Aggregate Queries on Outsourced Databases” In Proceedings of Privacy Enhancing Technologies in Springer, Lecture Notes in Computer Science Volume 5672, 2009, pp 185-201.

[19] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in S&P’07. IEEE Computer Society, 2007, pp. 321–334.

[20] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization,” in PKC’11. Springer, 2011, pp. 53–70.

[21] Kan Yang, Xiaohua Jia, Kui Ren, Bo Zhang ,“DAC-MACS: Effective Data Access Control for Multi-Authority Cloud Storage Systems” In Proceedings of IEEE INFOCOM, 2013, pp. 2895-2903.

[22] Hue T. B. P., Luyen G. N., Kha N. D., S. Wohlgemuth, “ An Efficient Fine-grained Access Control Mechanism for Database Outsourcing Service”, In Proc. of IEEE International Conference on Information Security and Intelligence Control (ISIC), 2012 , 65-69.

[23] M. Xie, H. Wang, J. Yin, and Meng, “Providing *freshness* guarantees for outsourced databases,” in Proceedings of the 11th international conference on Extending database technology: Advances in database technology, ser. EDBT ’08. New York, NY, USA: ACM, 2008, pp. 323–332.

[24] P. Ghazizadeh, R. Mukkamala S. Olariu, “Data Integrity Evaluation in Cloud Database-as-a-Service”, In Proceedings of IEEE Ninth World Congress on Services, 2013, pp. 280-285.

[25] V. Ciriani, S. D. Vimercati, S. Foresti, and S. Jajodia, "Combining Fragmentation and Encryption to Protect Privacy in Data Storage," *TISSEC*, vol. 13, pp. 1094-9224, 2010.

[26] V. Ciriani, S. D. C. D. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Fragmentation and Encryption to Enforce Privacy in Data Storage," *ESORICS 2007*, pp. 171-186