ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

Securing Data in Document Oriented Database MongoDB

Kusum Kakwani[1], Naziya Pathan[2] , Shyam Dubey[3], I.C. Mehta[4]

M. Tech. Scholar, Department of CSE., Nuva College Of Engineering And Technology, Nagpur (MS), India.[1]

Asst. Professor, Department of CSE., Nuva College Of Engineering And Technology, Nagpur (MS), India. [2, 3]

Associate Professor, Department of Computer Technology, MIET, Gondia (MS), India. [4]

ABSTRACT: Space, Time and Privacy have become a key requirement for database management systems. NoSQL datastores, namely highly compress data on non relational database management systems, which often support data management of Internet applications, still do not provide support for privacy preservation. In this paper, an approach is used on the basis of purpose and role based policies for preserving privacy in document oriented database MongoDB. It consists of the enhancement of the MongoDB level based access control model with privacy keys for security and monitor. The proposed monitor is easily used into any MongoDB deployment control with high protection for data security.

KEYWORDS: Document Oriented, MongoDB, NoSQL datastores, Privacy, Purpose-based access control

I. INTRODUCTION

NoSQL datastores are emerging non relational databases committed to provide high scalability for database operations over several servers. These platforms are getting increasing attention by companies and organizations for the ease and efficiency of handling high volumes of heterogeneous and even unstructured data. Although NoSQL datastores can handle high volumes of personal and sensitive information, up to now the majority of these systems provide poor privacy and security protection. Initial research contributions started studying these issues, but they have mainly targeted on security aspects. To the best of our knowledge, we are not aware of any work targeting privacy-aware access control for NoSQL systems, but we believe that privacy policies can also be enforced in NoSQL database systems similar to what has been proposed for Relational Database Management Systems. With this work, we begin to solve this issue, by proposing an approach using purpose-based access control capabilities into MongoDB, one of the most popular NoSQL datastore. However, different from relational databases, where all existent systems refer to the same data model and query language, NoSQL datastores operate with various languages and data models. This variety makes the definition of a general approach to have privacy-aware access control into NoSQL datastores a very important goal. It is believed that a stepwise approach is necessary to define such a general solution. As such, here, focus is on: 1) a single datastore, and 2) selected rules for privacy policies. The problem is approached by focusing on MongoDB, which, according to the DB-Engines Ranking, 2 ranks, by far, as the most popular NoSQL datastore. MongoDB uses a document-oriented data model. Data are modelled as documents, namely records, possibly images collections that are stored into a database[1].

Analysis has been done on several privacy-aware access control models proposed for relational DBMSs to identify the characteristics of privacy policies to be supported. In all the analysed models [2]:

1) Privacy policies require rule based and enforcement mechanisms, as different data owners can have different privacy requirements on their data.

2) The purposes for which data should be accessed with those for which they are stored are considered as the key required condition to grant the access and thus it is important for any privacy policy.

DOI: 10.15680/IJIRCCE.2017. 0503165

As such, fine grained purpose-based policies have been selected as the target policy type for our proposal. MongoDB integrates a role-based access control (RBAC) model which supports user and role management, and enforces access control at collection level. However, no support is provided for purpose-based policies. As such, this paper extends MongoDB RBAC with the support for purpose-based policy specification and enforcement at document level. More precisely, the rule level at which the MongoDB RBAC model operates, integrating the required support for purpose related concepts. On top of this enhanced model an efficient enforcement monitor, called Mem (MongoDB enforcement monitor), which has been designed to operate in any MongoDB deployment. Within the client/server architecture of a MongoDB deployment, a MongoDB server front-end interacts, through message exchange, with multiple MongoDB clients. Mem operates as a proxy in between a MongoDB server and its clients, monitoring and possibly altering the flow of messages that are exchanged by the counterparts[3].

Access control is enforced by means of MongoDB message rewriting. More precisely, either Mem simply forwards the intercepted message to the respective destination, or injects additional messages that encode commands or queries. In case the intercepted message encodes a query, Mem writes it in such a way that it can only access documents for which the specified policies are satisfied. The integration of Mem into a MongoDB deployment is straightforward and only requires a simple configuration. No programming activity is required to system administrators. Additionally, Mem has been designed to operate with any MongoDB driver and different MongoDB versions. First experiments conducted on a MongoDB dataset of realistic size have shown a low Mem enforcement overhead which has never compromised query usability.

MongoDB : Databases, Schemas and Tables

Databases: MongoDB is a document-oriented DBMS, with JSON-like objects comprising the data model, rather than RDBMS tables. MongoDB does not support joins and transactions. However, it features secondary indexes, an expressive query language, atomic writes on a per-document level, and fully-consistent reads. MongoDB uses BSON, a binary object format similar to, but more expressive than JSON.

Schemas: MongoDB uses dynamic schemas. We can create collections without defining the structure, i.e. the fields or the types of their values, of the documents. You can change the structure of documents simply by adding new fields or deleting existing ones. Documents in a collection need unique set of fields.

Tables: MongoDB database stores its data in collections not in tables. The collections are the rough equivalent of RDBMS tables. A collection holds one or more documents, which corresponds to a record or a row in a relational database table, and each document has one or more fields, which corresponds to a column in a relational database table.

MongoDB allows clients to read documents inserted or modified before it commits these modifications to disk, regardless of write concern level or journaling configuration. Applications may observe two classes of behaviours:

- MongoDB will allow clients to read the results of a write operation before the write operation returns for systems with multiple concurrent readers and writers
- If the MongoDB terminates before the journal commits, even if a write returns successfully, queries may have read data that will not exist after the MongoDB restarts.

Other database systems refer to these isolation semantics as read uncommitted. For all inserts and updates, MongoDB modifies

- each document in isolation
- clients never see documents in intermediate states

For multi-document operations, MongoDB does not provide any multi-document transactions or isolation. When MongoDB returns a successful journal write concern, the data is fully committed to disk and will be available after

MongoDB restarts. For replica sets, write operations are durable only after a write replicates and commits to the journal of a majority of the voting members of the set.

II. RELATED WORK

In [9], Authors proposed an approach for privacy preserving access control based on the notion of purpose within Relational database management systems. Purpose information associated with a given data element specifies the intended use of the data element. A key feature is that it allows multiple purposes to be associated with each data element and also supports explicit prohibitions, thus allowing privacy officers to specify that some data should not be used for certain purposes. An important issue addressed in this article is the granularity of data labelling, that is, the units of data with which purposes can be associated. This issue is addressed in the context of relational databases based on SQL.

Big Data platforms allow the storage and analysis of high volumes of data with heterogeneous format from different sources. Big Data platforms have been specifically designed to support advanced form of analytics satisfying strict performance and scalability requirements. However, no proper consideration has been devoted so far to data protection. Big Data platforms integrate quite basic form of access control, and no support for privacy policies. Although the potential benefits of data analysis are manifold, the lack of proper data protection mechanisms may prevent the adoption of Big Data analytics by several companies. This motivates the fundamental need to integrate privacy and security awareness into Big Data platforms. In [10], Authors defined a framework that supports the integration of privacy aware access control features into existing Big Data platforms.

Privacy of data owners and query users is vital in modern cloud data management. In [12], Authors proposed a solution of data storage and query protocol based on classical homomorphic encryption. The scheme is given to preserve privacy of both data owners and query users. Main efforts are put on NoSQL database which is less structural than relational database. Storage and indexing structure on NoSQL database, query protocol are proposed, and algorithms for updating and querying are also given. To implement it, Berkley DB, an excellent storage solution for NoSQL database is chosen and data are encrypted/decrypted using Elgamal and Paillier encryption system.

In [13] Authors presented a role-involved conditional purpose-based access control (RCPBAC) model, where a purpose is defined as the intension of data accesses or usages. RCPBAC allows users using some data for certain purpose with conditions. An algorithm is developed to achieve the compliance computation between access purposes and intended purposes and is illustrated with role-based access control (RBAC) to support RCPBAC. According to this model, more information from data providers can be extracted while at the same time assuring privacy that maximizes the usability of consumers' data. It extends traditional access control models to a further coverage of privacy preserving in data mining environment as RBAC is one of the most popular approach towards access control to achieve database security and available in database management systems. The structure helps enterprises to circulate clear privacy promise, to collect and manage user preferences and consent.

Nowadays, many research institutions and IT industries are storing their data on the cloud. On the other hand the controllers/owners are worried about their data that is stored on the cloud which is not secure. In [19], Authors proposed a method to secure the data from attackers who are trying to access the data in an unauthorized way in MapReduce Systems, Honeypots are spread over the data which is only accessed by the attackers and/or unauthorized users but not by authorized MapReduce jobs. The analysis indicates that the detection of unauthorized data access with satisfactory performance could be found in MapReduce based cloud environments.

III. EXISTING SYSTEM

As the size of database is getting bigger, tuning performance of databases is getting more importance to Researchers, enterprises and database administrators. Selection of an appropriate set of indexes for the workload processed by the database system is main part of physical design and performance tuning. Solution to the index selection problem using Mining Index Selection Approach relies in the combination of [5]:

- Recommendation of index type for proposed indexes
- Using frequent itemset as a method to build a certain order of combined indexes out of fields for each frequent query

- Use of query optimizer to select the final recommended indexes
- The approach is to create virtual indexes which remove any alteration in the database
- Applying the approach to a document-oriented NoSQL database MongoDB

It provides an outline for developing an automatic index selection system and reduces administrator’s overhead. The typical setting involves two users: one that gets information from the other that is to share (only) the requested information. Consequently, there is a tension between information sharing and privacy. On the one hand, sensitive data needs to be kept confidential; on the other hand, data owners may be willing, or forced, to share information. NoSQL datastores are highly scalable and provides improved performance but there is lack of support for data security.

IV. SOFTWARE DEVELOPMENT LIFE CYCLE AND REQUIREMENT ANALYSIS

The System Development Life Cycle is the process of developing information systems through investigation, analysis, design, implementation, and maintenance. The System Development Life Cycle (SDLC) is also known as Information Systems Development or Application Development [4]. Below are the steps involved in the System Development Life Cycle. Each phase within the overall cycle may be made up of several steps.

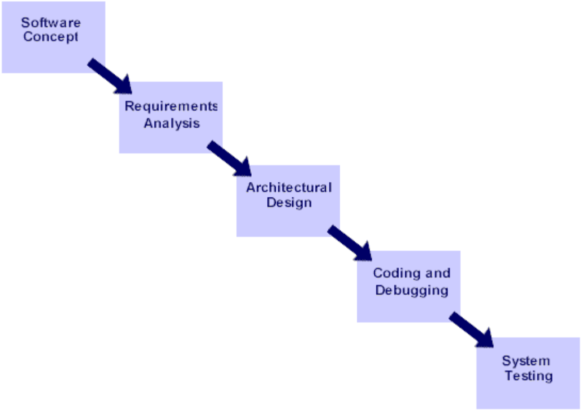

Figure 1. System Development Life Cycle

In the analysis phase the selected processes are analysed with a view to goals set. Following requirements were identified for building the system:

- To improve and accelerate learning.
- To enhance team collaboration & coordination.
- To maximize the organizations’ use of available collective wisdom, experience, and the brain-power of human capital assets.
- In general, be better able to Create, Capture, Share, Protect, Disseminate, Package and Exploit knowledge, intellectual capital, and intangibles.
- Help to create a more adaptive, responsive, dynamic & flexible organization.
- To unleash new ideas and creativity.
- To lift productivity and efficiency.
- To innovate speed.

V. ANALYSIS OF PROPOSED SYSTEM

The general approach to the rule of privacy-aware access control into NoSQL data stores is a very important goal. Users are only allowed to execute for access purposes for which they have a proper authorization. Purpose authorizations are granted to users as well as to roles. The data storage and network transfer format for documents, simple and fast. The MongoDB RBAC model is characterized by the concepts of privilege, data resource, action, role, and user. It regulates the access to document collections on the basis of the privileges granted to roles. We enhance this basic scheme introducing fine grained purpose-based access control at document level. A privilege models a set of actions that access a data resource. The referred resource can be a collection, a set of collections, a database, or a set of databases, whereas actions are a subset of the predefined MongoDB data management operations (e.g., find, update, remove, and insert), administrative and review functions (e.g., add/remove a user/role, show users/roles).A role models a set of privileges to be assigned to users. Roles definition is aligned with the principles of the proposed NIST standard for RBAC [6]. Roles are hierarchically organized and a role can extend other roles inheriting their privileges. The set of privileges that are granted to a role r is the union of the privileges specified for r and those specified for each role from which r descends. MongoDB includes a set of predefined roles and allows administrators to introduce custom roles by means of administrative functions. Finally, element user models a stakeholder that interacts with MongoDB requiring the execution of data manipulation operations, administrative, or review functions. When a role is assigned to a user, the user receives the authorization to execute all the actions specified by the privileges associated with the role on the specified data resources. We have enhanced the MongoDB RBAC model with additional conceptual elements, instrumental to support purpose-based access control. The key element of the enhanced model is the concept of purpose, which is used to specify the reasons for which documents can be accessed, and to declare the reasons for which users aim at accessing them. The union of the purposes considered for an application scenario forms the scenario purpose set Ps. The purposes for which a document can be accessed are denoted as intended purposes. The intended purpose set specified for a document d is a set Ip that groups the identifiers of the purpose elements of Ps which specify the reasons for which the access to d is allowed. Data mining discovers hidden relationships in data, in fact it is part of a wider process called "knowledge discovery". Knowledge discovery describes the phases which should be done to ensure reaching meaningful results. Using data mining tools does not completely eliminate the need for knowing business, understanding the data, or familiarity with statistical methods. It also does not include clear patterns of knowledge. Data mining activities are divided into three categories:

1. Discovery: Includes the process of searching the Database to find hidden patterns without a default Preset.
2. Predictive Modelling : Includes the process of discovering patterns in databases and use them to Predict the future.
3. Forensic Analysis: Includes the process of applying extracted patterns to find unusual elements.

It is observed that the decentralized approach introduces performance overheads due to the additional read and write requests to the global storage for acquiring and releasing locks. Therefore, projected an alternative approach of using a dedicated service for conflict detection. In the service-based approach, the conflict detection service maintains in its primary memory the information required for conflict detection. A transaction in its commit phase sends its read/write sets information and snapshot timestamp value to the conflict detection service. We designed this service to support conflict detection for the basic-SI model and the cycle prevention/detection approaches for serializable transaction. Based on the particular conflict detection approach, the service checks if the requesting transaction conflicts with any previously committed transaction or not. If no conflict is found, the service obtains a commit timestamp for the transaction and sends a 'commit' response to the transaction process along with the commit timestamp, otherwise it sends 'abort'. Before sending the response, the service updates the transaction's commit status in the Transaction Table in the global storage.

VI. DISK ENCRYPTION

There are multiple ways to encrypt data at rest with MongoDB. Encryption can be implemented at the application level, or via external file system and disk encryption solutions. By introducing additional technology into the stack, both of these approaches can add cost and complexity. With the introduction of the Encrypted storage engine in MongoDB 3.2, protection of data at-rest becomes an integral feature of the database. By natively encrypting database files on disk, administrators eliminate both the management and performance overhead of external encryption mechanisms. This new storage engine provides an additional level of defence, allowing only those staff with the appropriate database credentials access to encrypted data. The storage engine encrypts each database with a separate key. The key-wrapping scheme wraps all of the individual internal database keys with one external master key for each server. The Encrypted storage engine supports two key management options in both cases; the only key being managed outside of MongoDB is the master key:

- Local key management via a key file.
- Integration with a key management.

Most regulatory requirements mandate that the encryption keys must be rotated and replaced with a new key. When using database files themselves to be re-encrypted, thereby avoiding the significant performance overhead imposed by key rotation in other databases. Only the master key is rotated, and the internal database key store is re-encrypted.

VII. CONCLUSION

Purpose concepts and related give mechanisms to regulate the access at document level on the basis of purpose and key based policies. An enforcement monitor, called Mem, has been designed to implement the proposed security. Mem operates as a proxy between MongoDB user and a MongoDB server, and enforces access control by monitoring and possibly manipulating the flow of exchanged messages. Furthermore, the presented approach can be generalized to support for multiple NoSQL datastores.

REFERENCES

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic databases. In28th International Conference on Very Large Data Bases(VLDB), 2002.
[2] K. Browder and M. A. Davidson. The Virtual Private Database in Oracle9iR2. Technical report, 2002. Oracle Technical White Paper.
[3] J. Byun and N. Li. Purpose based access control for privacy protection in relational database systems.The VLDB Journal, 17(4),2008.
[4] R. Cattell. Scalable SQL and NoSQL Data Stores. SIGMOD Rec., 39(4):12–27, May 2011.
[5] Parinaz Ameri ,Jorg Meyer and Achim Streit "On a New Approach to the Index Selection Problem using Mining Algorithms" 2015 IEEE International Conference on Big Data (Big Data)
[6] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS), 26(2):4, 2008.
[7] P. Colombo and E. Ferrari. Enforcement of purpose based access control within relational database management systems. IEEE Transactions on Knowledge and Data Engineering (TKDE), 26(11), 2014.
[8] P. Colombo and E. Ferrari. Enforcing obligations within relational database management systems. IEEE Transactions on Dependable and Secure Computing (TDSC), 11(4), 2014.
[9] P. Colombo and E. Ferrari. Efficient enforcement of action aware purpose-based access control within relational database management systems. IEEE Transactions on Knowledge and Data Engineering, 27(8), 2015.
[10] P. Colombo and E. Ferrari. Privacy aware access control for big data: A research roadmap. Big Data Research, 2015. http://dx.doi.org/10.1016/j.bdr.2015.08.001.
[11] D. Ferraiolo, R. Sandhu, S. Gavrila, D. Kuhn, and R. Chandramouli.Proposed NIST standard for role-based access control. ACM Transactions on Information and System Security (TISSEC), 4(3), 2001.
[12] Y. Guo, L. Zhang, F. Lin, and X. Li. A solution for privacypreserving data manipulation and query on NoSQL database. Journal of Computers, 8(6):1427–1432, 2013.
[13] M. Kabir, H. Wang, and E. Bertino. A role-involved conditional purpose-based access control model. In M. Janssen, W. Lamersdorf, J. Pries-Heje, and M. Rosemann, editors, E-Government, Eservices and Global Processes, volume 334 of IFIP Advances in Information and Communication Technology, pages 167–180. Springer Berlin Heidelberg, 2010.
[14] M. E. Kabir and H. Wang. Conditional purpose based access control model for privacy protection. In ADC 2009.
[15] B. Klimt and Y. Yang. The Enron corpus: a new dataset for email classification research. In Machine learning: ECML 2004.
[16] D. Kulkarni. A fine-grained access control model for key-value systems. In Proceedings of the third ACM conference on Data and application security and privacy, pages 161–164. ACM, 2013.
[17] N. Leavitt. Will NoSQL databases live up to their promise? Computer, 43(2), Feb 2010.
[18] Q. Ni, E. Bertino, J. Lobo, C. Brodie, C. Karat, J. Karat, and A. Trombetta. Privacy-aware role-based access control. ACM Transactions on Information and System Security (TISSEC), 13(3), 2010.
[19]H. Ulusoy, P. Colombo, E. Ferrari, M. Kantarcioglu, and E. Pattuk. GuardMR: fine-grained security policy enforcement for MapReduce systems. In ACM ASIACCS 2015.
[20] A. Cavoukian. Privacy by Design: leadership, methods, and results. In S. Gutwirth, R. Leenes, P. de Hert, and Y. Poullet, editors, European Data Protection: Coming of Age. Springer, 2013.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 3, March 2017

BIOGRAPHY

Kusum Kakwani is pursuing M. Tech. degree in Computer Science and Engineering from Nuva college of Engineering, RTM Nagpur University, Nagpur, India. She has teaching experience of more than 6 years in academic field. Her areas of interest are Data Mining, DBMS, Security and Theory of Computer Science.

DOI: 10.15680/IJIRCCE.2017. 0503165