SECURITY ANALYSIS OF UNSTRUCTURED DATA IN NOSQL MONGODB DATABASE

Jitender Kumar
M.Tech Scholars
Jaypee Institute of Information Technology
Jitenderkumar2929@gmail.com

Varsha Garg
Assistant Professor
Jaypee Institute of Information Technology
Varsha.garg@jiit.ac.in

Abstract: - NoSQL databases systems are non-relational databases uniquely intended to give high accessibility, reliability, and scalability for enormous data. Additionally sharding is the main fundamental favorable circumstances of NoSQL database. Various companies are moving towards NoSQL databases. NoSQL databases can store unstructured data such as email, multimedia, documents, and social media with high performance. NoSQL document stored database, MongoDB has many security risks which can be overcome by a good secure cryptographic system.

In this paper, we will use symmetric cryptographic techniques for providing the security (confidentiality) of unstructured data in NoSQL document stored MongoDB. DES, AES, and blowfish algorithms with random key generation are used to encrypt/decrypt the document data before storing/retrieving to/from the NoSQL MongoDB database. We have also provided the comparative analysis of execution time taken by each algorithm with MongoDB for different size of data. There arises a problem that the storage size taken by the encrypted data in MongoDB database is more as compared to the original data. To solve this problem, we used a zlib compression technique to reduce the storage size taken by the encrypted data and provide comparative results.

Key words: NoSQL, Unstructured Data, MongoDB, DES, AES, Blowfish

I INTRODUCTION

In the last few years enormous data (structured, semi-structured and unstructured) is growing very fast and has become more complex with respect to the 3V's (volume, variety, and velocity). Big data is an extensive or complex set of data. Relational data processing programs or applications are insufficient to manage it. To overcome this problem, NoSQL (Not only SQL) databases are designed. NoSQL database are non-relational DBMS (which may or may not support a querying language). NoSQL databases can deal with organized and unstructured enormous data. There is no fixed schema; Enormous data can be stored in the form of document oriented datastores, key-value datastores, column family datastores and graph datastores [3].

A. MongoDB Concepts

In MongoDB, there are no database outlines or tables. Documents are like rows and are assembled into collections which are like tables. The document is an information structure made out of field and value pairs. The value of fields may incorporate different records, clusters, and varieties of documents. MongoDB consequently produces a primary key (id) to uniquely recognize each record. MongoDB endeavors to hold the greater part of the information in memory so straightforward questions take less time by staying away from costly hard disk recovery operations.

978-1-5386-0627-8/17/$31.00 ©2017 IEEE

II RELATED WORK

MongoDB has file storage system, data files are unencrypted in this, and it doesn’t provide any encryption scheme to automatically encrypt/decrypt these datasets. It means that any active or passive attacker or unauthorized user can easily access the file system and retrieve the valuable or secret information. To reduce this, the application should explicitly encrypt/decrypt any confidential data before updating/retrieving into/from the databases [11][12].

The authors of [16] explain the diverse components of database management system, to be specific Flat File Database, Relational DBMS and NoSQL were evaluated. The fundamental issues on the Flat record and Relational DBMS is that both were database systems incorporate security issues, scalability issues, and timely propagation of changes to guarantee consistency, accessibility of information, though paying little heed to the network segment. Relational and file record system were not able to handle the enormous data generated by the interactive applications. Thus organizations are moving towards NoSQL datastores adoption. NoSQL datastores systems support the high scalability and provide the high performance inherent in traditional DBMSs. In this article, author also discussed the different database systems and different properties/theorems namely ACID, BASE and CAP theorems respectively, are evaluated.

In [11], authors provide the explanation of security problems and features of two NoSQL databases Cassandra and MongoDB. Security feature of NoSQL MongoDB and Cassandra are as: 1) data at rest is unencrypted; 2) weak authorization, 3) no authentication by default and 4) no auditing of data. Though Inter-network communication encryption option is available but not by default. Also client communication is not available resulting in MongoDB being vulnerable to an injection attack.

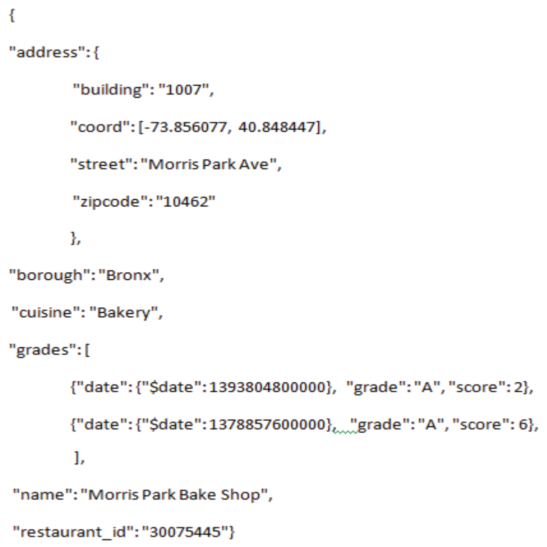
In [17], authors have shown that NoSQL (MongoDB) datastores are also vulnerable to injection attacks as the SQL based database system and one approach to prevent these is through careful code analysis and/or static analysis. But it may have high false positive rates and presents difficulty for read. While dynamic analysis tools/methods appeared to be extremely valuable for the identification/detection of injection attacks [18], these should be changed as per recognize the specific vulnerabilities of NoSQL databases that they portrayed in this paper.

In [20], authors used various symmetric key cryptographic schemes for encrypting the data of MongoDB at application level but they have not defined the key size of these algorithms used by them. So in this paper, we will use DES-64, AES-128, and Blowfish-64 algorithms with different key sizes and key will be generated randomly. AES [9], DES [8], and blowfish [6] are symmetric cryptographic algorithms used to encrypt/decrypt the information before storing/retrieving to/from the NoSQL MongoDB database [7][10].

We also provided the security analysis of execution time taken by each algorithm with MongoDB for different size of data. There occurs a problem that storage size taken by encrypted data in MongoDB database is much more than the unencrypted data, to solve this problem, we used a Zlib compression technique to reduce the space taken by encrypted data.

III DESCRIPTION AND METHODOLOGY

In this article, we used the restaurant unstructured data [15] for analyzing the time complexity of different cryptographic techniques with MongoDB for different size of data. The restaurant data is in the form of JSON [3] format. The following structure is based on '***restaurants'*** collection. Structure of 'restaurants' collection is as under:



```
{
"address": {
        "building": "1007",
        "coord": [-73.856077, 40.848447],
        "street": "Morris Park Ave",
        "zipcode": "10462"
        },
"borough": "Bronx",
"cuisine": "Bakery",
"grades": [
        {"date": {"$date": 1393804800000},  "grade": "A", "score": 2},
        {"date": {"$date": 1378857600000},  "grade": "A", "score": 6},
        ],
"name": "Morris Park Bake Shop",
"restaurant_id": "30075445"}
```

Figure 1: Unstructured Document data structure [15].

We design a secure architecture for achieving data confidentiality in MongoDB, by using different symmetric encryption standard as shown in figure 2. Confidential data is encrypted before storing in database and decrypted after accessing from the database.

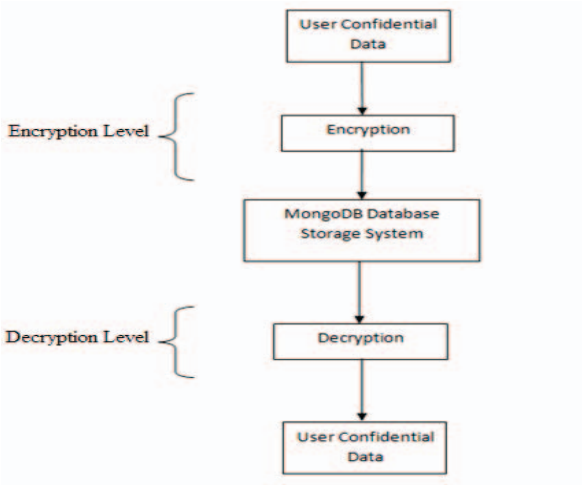


Figure 2: Encryption/Decryption Process

IV RESULTS AND ANALYSIS

As we have discussed earlier, NoSQL MongoDB has many security vulnerabilities. To overcome these vulnerabilities we used different cryptographic systems and analyze their time complexity. We used the three different symmetric cryptographic techniques named as AES [9], DES [8], and Blowfish [6][7]. AES is much more secure than others because the key length of AES is 128 bit based on substitution and permutation network but different types of attack can be possible on Blowfish and DES.

Encryption and decryption execution time is taken in average for each set after running the code 10 times of inserting/retrieving the data into/from the MongoDB database.

Table 1 : Encryption/decryption time of AES-128, DES-64, and Blowfish-64.

Size KB	Data Set	AES (128)		DES (64)		Blowfish (64)	
		Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)	Encryption (ms)	Decryption (ms)
365	SetA	139	97	153	119	43	31
866	SetB	212	153	292	204	69	46
1207	SetC	249	190	389	284	98	53
1582	SetD	298	246	449	303	108	73
1984	SetE	354	286	505	362	139	87

Table 1 shows the comparative study of AES, DES, and Blowfish with MongoDB. It shows the encryption and decryption process time of different data set using AES, DES, and Blowfish. The total execution time of encryption and decryption of all these cryptographic systems will be calculated respectively to analyze these systems suitability with NoSQL MongoDB database. So to enhance the security mechanism we have to apply the faster cryptographic algorithm in order to save the computational power. If required time is decreased then power consumption also decreases. Here Blowfish execution time is least followed by AES and then DES in all cases..

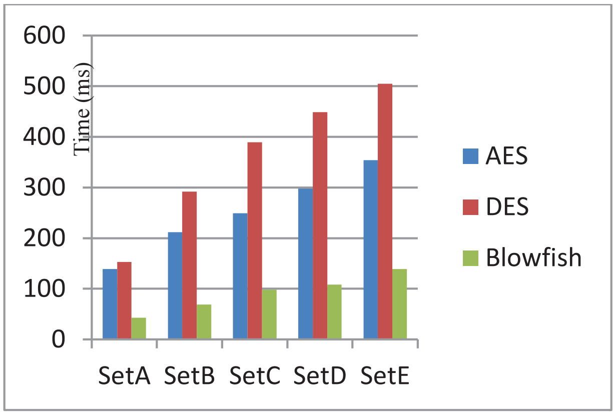


Figure 3: Encryption Execution time of AES, DES and Blowfish

Figure 3, represents the graph which shows the time taken in encryption process for different data set by different symmetric key encryption standards before inserting the data in MongoDB database. This graph clearly indicates that Blowfish technique requires less time for encryption process for different size of data and AES comparatively performs better than DES for all data set. This graph also shows that performance increases as time decreases and vice versa.

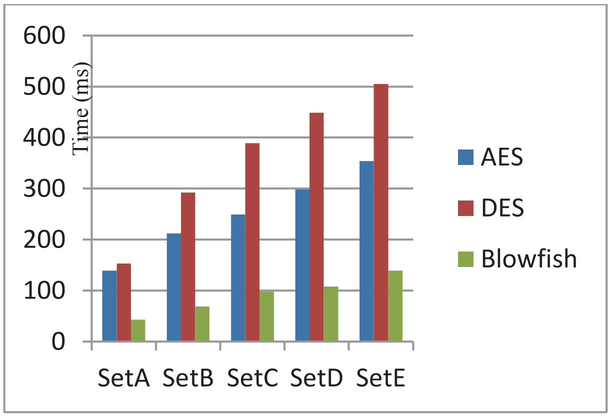


Figure 4: Decryption Execution time of AES, DES Blowfish

Figure 4, represents the graph which shows the time taken in decryption process for document oriented dataset by different symmetric key encryption systems after retrieving from MongoDB database. This graph clearly indicates that AES technique required less time for decryption process than DES and more time than blowfish. Blowfish required less time to decrypt the same size of data than AES and DES.

We can encrypt the data using different cryptographic techniques which may secure the confidential information, based on the strength and weakness of the encryption algorithm but there is a problem that if we encrypting the database then the size taken by the encrypted information is greater than unencrypted data.

Table 2 shows that the size taken by the encrypted information is greater than the unencrypted data but same for all used cryptographic system in MongoDB database. Encryption algorithm doesn't matter for storage required for encrypted data.

Table 2: Space taken by encrypted data for AES, DES, and Blowfish

Size KB	Data Set	AES (128)	DES (64)	Blowfish (64)
		Encrypted Data Size (KB)	Encrypted Data Size (KB)	Encrypted Data Size (KB)
365	SetA	528	528	528
866	SetB	1236	1236	1236
1207	SetC	1720	1720	1720
1582	SetD	2248	2248	2248
1984	SetE	2816	2816	2816

There is a problem that the encrypted data requires more storage space than the original (unencrypted) dataset. To solve this problem, we used a zlib compression [19] technique to reduce the storage size taken by the encrypted data.

Table 3: zlib compression on encrypted data

Size KB	Data Set	Encrypted Data at Storage (KB)	zlib Compression (KB)	Compressed (%) of Encrypted data	Compressed (%) of Original (Unencrypted)data
365	SetA	528	364	31.06 %	0.27 %
866	SetB	1236	840	32.03 %	3.00 %
1207	SetC	1720	1168	32.09 %	3.23 %
1582	SetD	2248	1528	32.02 %	3.41 %
1984	SetE	2816	1912	32.10 %	3.62 %

Table 3 clearly shows that zlib compression technique reduces the space required for storing encrypted data in MongoDB database. Zlib compression technique is applied for different data set and the results are compared with the unencrypted and encrypted data. Table 3 also shows the percentage of data storage reduced with respect to unencrypted and encrypted data set in mongodb database.

V EXPERIMENT SET UP

The whole set up is implemented in Window 7 platform using Java 8 language, Net beans and MongoDB running on Intel core i3 (2.10GHz) machine with 3GB RAM. The performance is tested on different sizes of data. As can be seen from Table 1, results obtained by blowfish technique are consistently better than the results by AES and DES for different size of data. It can be inferred from the result that the time consumed to encrypt and decrypt the data is less in AES as compare to DES and more as compare to Blowfish.

VI CONCLUSION AND FUTURE WORK

Most popular NoSQL MongoDB database has various security issues. The main problem of this system is that it does not support the encryption/decryption by default and is vulnerable to injection attack and has exposure to DOS attacks. To overcome these security issues in NoSQL we used the concept of symmetric-key cryptography such as advanced encryption standard, data encryption standard, and blowfish for encrypting/decrypting the data for NoSQL document-oriented MongoDB.

This paper provides a better scheme for encoding the data for document-oriented MongoDB. These cryptographic algorithms are used different keys lengths for encrypting/decrypting the data which overcome the problems of NoSQL MongoDB database. A tradeoff exists amongst security and proficiency for data encryption/ decryption time analysis and the key size used by them for providing the better security. As we find that Blowfish encryption/decryption algorithm is giving better performance than AES and DES. Different attacks can be possible on DES and Blowfish like brute force attack, differential attack, and linear attack. AES is most suited to apply to the client-server architecture in NoSQL MongoDB because it provides better security mechanism than other algorithms.

Clearly, the future eras of NoSQL database require extensive advancement and hardening in the request to give a safe environment to private data which is being secured by applications, (for example, informal communities) utilizing them.

So, in future, we will try to achieve the confidentiality and integrity simultaneously.

We will also apply the efficient encryption algorithm in SSL/SSH to give a digital certificate in order to achieve good security level and overcome the flaws in NoSQL MongoDB. The performance of the NoSQL MongoDB can be improved by only encrypting the sensitive data field.

REFERENCES

[1] Moniruzzaman, A. B. M., and Syed Akhter Hossain, "NoSQL database: New era of databases for big data analytics-classification, characteristics and comparison", International Journal of Database Theory and Application, Vol. 6, 2013.

[2] Hecht, Robin, and Stefan Jablonski, "Nosql evaluation", International conference on cloud and service computing. IEEE, 2011.

[3] Gómez, Paola, Rubby Casallas, and Claudia Roncancio, "Data schema does matter, even in NoSQL systems!, Research Challenges in Information Science (RCIS), 2016 IEEE Tenth International Conference on. IEEE, 2016.

[4] Tauro, Clarence JM, S. Aravindh, and A. B. Shreeharsha, "Comparative study of the new generation, agile, scalable, high performance NOSQL databases", International Journal of Computer Applications 48.20: pp.1-4 2012.

[5] Tilmann Rabl, Mohammad Sadoghi, HansArno Jacobsen, "Solving big data challenges for enterprise application performance management", Proceedings of the VLDB Endowment 5.12, pp.1724-1735 Vol. 5, 2012.

[6] Milind Mathur, Ayush Kesarwani, "comparison between des, 3des, rc2, rc6, blowfish and aes", Proceedings of National Conference on New Horizons in IT - NCNHIT 2013.

[7] Diaa Salama Abdul, Elminaam, Hatem Mohamed Abdul Kader and Mohie Mohamed Hadhoud, "Performance Evaluation of Symmetric Encryption Algorithms", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.12, December 2008.

[8] Coppersmith, Don, "The Data Encryption Standard (DES) and Its Strength Against Attacks", IBM Journal of Research and Development, pp. 243 - 250 May 1994.

[9] Daemen, J, and Rijmen, Vincent Rijmen "Rijndael: The Advanced Encryption Standard", Dr. Dobb's Journal, March, PP. 137-139, 2001.

[10] Mongodb, http://www.mongodb.org/.

[11] Nurit Gal-Oz, Yaron Gonen, "Security issues in nosql databases" Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on. IEEE, 2011.

[12] Erez Shmueli, Ronen Vaisenberg, Yuval Elovici, Chanan Glezer, "Database Encryption – An Overview of Contemporary Challenges and Design Considerations", Proceeding of the 2009 ACM SIGMOD International Conference on Management of Data, Vol. 38, No. 3, September 2009.

[13] Oracle Databases, http://www.oracle.com/databases/overview

[14] Brewer's CAP Theorem, http://www.julianbrowne.com/viewer/brewers-cap-theoram.

[15] Restaurant Data Set, https://raw.githubusercontent.com/mongodb/docs-assets/primer-dataset/primer-dataset.json

[16] Innocent Mapanga, Prudence Kadebu, "Database Management Systems: A NoSQL Analysis", International Journal of Modern Communication Technologies & Research (IJMCTR) ISSN: 2321-0850, Volume-1, Issue-7, September 2013.

[17] Ron, Aviv, Alexandra Shulman-Peleg, and Emanuel Bronshtein, "No SQL No Injection?, Examining NoSQL Security", In Proceedings of the 9th Workshop on Web 2.0 Security and Privacy (W2SP), Vol. 1, *2015*.

[18] Ron, Aviv, Alexandra Shulman-Peleg, and Anton Puzanov, "Analysis and Mitigation of NoSQL Injections", *IEEE Security & Privacy* 14.2: pp. 30-39, 2016.

[19] Compression in MongoDB, http://www.mongodb.com/blog/post/new-compression-options-mongodb-30

[20] Charmi Pariawala, Ravi Sheth, "Encrypting Data of MongoDB at Application Level", Advances in Computational Sciences and Technology Volume 10, Number 5 (2017) pp. 1199-1205.