A Comparison the Level of Security on Top 5 Open Source NoSQL Databases

Preecha Noiumkar
Mahasarakham University
6 PUBLICATIONS **34** CITATIONS

Thawatchai Chomsiri
Mahasarakham University
52 PUBLICATIONS **253** CITATIONS

A Comparison the Level of Security on Top 5 Open Source NoSQL Databases

Preecha Noiumkar, and Tawatchai Chomsiri

Abstract--**This research investigated the effective factors on the security of the top 5 open source NoSQL databases which are MongoDB, Cassandra, CouchDB, Hypertable, and Redis, focusing on 5 effective security factors including: (1) Data files encryption, (2) Client/Server Authentication/Encryption, (3) Inter-cluster Authentication/Encryption, (4) Script Injection, and (5) Denial of Service attacks. The researchers also compared the strength and weakness regarding these databases' security and found that Hypertables and Redis are the most secured databases to handle the attack launched by internet users (mostly with injection and DoS), and CouchDB, MongoDB, and Cassandra were the database that is mostly safe from packet sniffing during the communication between the servers. Surprisingly, these five databases did not perform the data file encryption. In this research, the researchers also suggested the useful methods to improve security level for these databases.**

Index Terms--**NoSQL Security, Database Security, Big Data, Cloud Computing**

I. INTRODUCTION

The rapid evolution of computer and internet has changed the way to manage the data. Many centralized databases have been changed to be distributed database s while the data's size becomes bigger. So, we can say that human has already entered an age of Big Data [1]. In the past, relational database such as Oracle, DB2, SQL server, and MySQL has offered the users with satisfactory services. However, after size of data and files has become bigger, e.g. VDO containing many Giga bytes, and DBMS' functioning that needs to support the clustering, these relational databases work slower [2], due to they need to work on the Cartesian operation between the relations in order to join tables and views altogether before selecting the suitable rows for the SQL command. According to these problems, new kinds of database has been introduced and it is called NoSQL database [3] which function more quickly than did the relational databases though it works in the Big Data environment and distributed environment. More importantly, many companies i.e. Facebook, Twitter, Amazon and Google [4] that store the data of 100 – 1,000 million members also use NoSQL databases. Currently, there are more than 100 brands of NoSQL databases [5]. Even Oracle, the giant database company, produces its own NoSQL products [6] and pushes them into the market. The NoSQL popularly used today is Open Source NoSQL. In this regard, many related organization and websites have ranked the NoSQL databases by its popularity and found similar result that the top five open source NoSQL databases are MongoDB, Cassandra, CouchDB, Hypertable, and Redis. Even though these databases function similarly or might work differently depending on the data types, one of the critical issues is security.

This research aims to investigate the effective factors on the security of these top 5 open source NoSQL databases in identify their strength and weakness and to offer the readers with some useful techniques for selecting the best NoSQL database that is mostly secured for their use in the network environment. The researchers conducted a comparative study on five effective security factors which are (1) Data files encryption, (2) Client/Server Authentication/Encryption, (3) Inter-cluster Authentication/Encryption, (4) Script Injection, and (5) Denial of Service attacks, and also analyzed the security vulnerabilities of each database in Section 3. Section 4 suggested the methods for making the databases more secured and Section 5 compared the security level of each database.

II. BACKGROUND

There are many organizations and websites that ranks the open source NoSQL databases by their popularity; for

Preecha Noiumkar is a lecturer at the Department of Information Technology, Faculty of Informatics, Mahasarakham University, Thailand (e-mail: preecha.n@msu.ac.th).

Thawatchai Chomsiri is an Assistant Professor at the Department of Information and Communication Technology in the Faculty of Informatics of the Mahasarakham University, Thailand. (e-mail: thawatchai@msu.ac.th).

ISBN:

example, a website 'www.bigdataanalyticsnews.com' [7] has ranked the top 12 open source databases in February, 2014; a website 'www.econsultancy.com' [8] has ranked the top five databases in April, 2012; a website 'www.nosql.findthebest.com' [9] has ranked the top 16 databases in year 2013; and a website 'fromdev.com' [10] has ranked the top 18 databases in year 2013. In particular, this research focused on the top five open source NoSQL databases ranked by a website 'www.bigdataanalyticsnews.com' [7] since it offers the mostly updated data. The NoSQL databases studied and analyzed in this research were MongoDB, Cassandra, CouchDB, Hypertable, and Redis, respectively.

MongoDB [11] is the database that supports the scale enlargement and functions very quickly. This database has been developed with C++ as a document oriented database supporting the full text functioning and is flexible for use. Cassandra [12] is the database with features of fault tolerance, high availability, and scalability. CouchDB [13] is a part of Apache project under JSON-based web application in which a user can use JavaScript to run MapReduce Queries on CouchDB. Hypertable [14] is the database that has been developed from the design of Google's BigTable and it can runs on distributed file system e.g. Apache Hadoop DFS, and it has been developed with C++. Finally, Redis [15] has its features of data structure that has been designed for a rapid functioning by applying an in-memory dataset, and it also supports several programming languages.

Each of these databases has its features of rapid functioning but still there has been a few research conducted on their security. Hence, the researchers decided to analyze the security factors of these top 5 open source NoSQL databases and the result were presented in the next section.

III. SECURITY VULNERABILITIES OF THE TOP 5 OPEN SOURCE NoSQL DATABASES

This section describes more about five types of the security vulnerabilities previously mention in Section 1 (Introduction) to indentify strength and weakness of each kind of database.

A. MongoDB

The data files of MongoDB are not encrypt automatically, so all data is stored as plain text. Thus, hackers can access the data directly and the data can be read immediately.

MongoDB typically supports the binary wire-level protocol by utilizing TCP port number 27017. Also, MongoDB has a feature called RESTful to manage its server via HTTP protocol on TCP port number 28017. However, there is no data encryption for these ports, which means that hackers can capture and get access to the data immediately since no data encryption was performed while the data was being sent back and forth between the client and server.

MongoDB does not support the authentication if it runs in sharded mode. Still, with either standalone mode or replica-set mode, the authentication can be activated on MongoDB in order to authenticate each server before joining the cluster. This authentication is based on the key or password called 'pre-shared secret' which has been hashed with MD5 algorithm before being stored in the key file. Therefore, when hackers get an access to the key file, they will see the hashed value of pre-shared secret, instead of plain text, in which the data becomes unusable for them. However, hackers can crack the pre-shared secret's hashed value by cracking the MD5.

In case of relational databases i.e. SQL server, Oracle, and MySQL, hackers can launch an attack on the database server or bypass the authentication on web servers by using the SQL injection. Similarly, hackers can attack MongoDB by using JavaScript. Samples of the attack launched on MongoDB are presented in [16].

Based on the data from www.cvedetails.com, it was reported that MongoDB version 2.4.0 – 2.4.4 always has a problem with the denial of service attack in which the remote authenticated users can perform the attack through the vulnerabilities of uninitialized pointer [17] and it makes the server crashed.

B. Cassandra

Cassandra's data files are stored without encryption and the database does not have an automatic data encryption, so hackers can get an access to the data which can be read immediately.

Cassandra utilizes Apache Thrift framework [18] to communicate with the client. The current version of Cassandra supports the SSL, but this SSL cannot be enabled by default so that the communication between the client and server becomes unencrypted.

Regularly, each node in the cluster communicates directly to one and another without the encryption and authentication. However, new versions of Cassandra (Ver. stable 0.8 branch or the upper one) will support the encryption between clusters.

Cassandra will manage Cassandra Query Language (CQL) so that DBA can simply and rapidly manipulate the data inside the database. This CQL has a similar syntax as SQL's, so it is believed that it can be attacked as same as the SQL injection. However, the CQL functions under Thrift [18] and in a level of RPC so it will not be attacked by the SQL injection. Although the study of Leonardo Aniello et. al. [19] suggested that Cassandra contains some vulnerabilities, those were not the CQL's.

Cassandra has a problem with the denial of service attack because it performs one thread per one client. Thus, if hackers created enough fake connections, they can make Cassandra lose its resources to those fake connections. If hackers knew the IP addresses of all Cassandra servers in the cluster and created enough fake connections, they can make Cassandra out of service. Additionally, Cassandra still has a problem in managing the inactive connections, since the timeout value is not set for the inactive connections.

C. *CouchDB*

CouchDB does not have an automatic data encryption as other NoSQLs, so the data files are in risk of being accessed and read directly. In term of the communication between the client and server or between the CouchDB servers, the authentication will be performed through a process called CRUD. In addition to the communication, there is also an encryption with a built-in SSL which is a component of CouchDB itself.

Kuon [20] previously proved that CouchDB contains vulnerabilities in the Script Injection since it uses JSON to manipulate the data, so hackers can launch JSON Injection to attack CouchDB.

A website www.securityfocus.com [21] sported that CouchDB also has the security vulnerabilities for the denial of service attack in which hacker can make it crashed and these vulnerabilities exist only on Apache CouchDB 1.5.0. Likewise, www.exploit-db.com [22] and www.redhat.com [23] suggested a new security vulnerability lately found on 03/24/2014, in which hackers can make the CouchDB servers crashed merely with a one-line command as follows.

```
curl http://couchdb_target/_uuids?count=9999999999999
99999999999999999999999999999999999
```

D. *Hypertable*

Similar to other NoSQLs, Hypertable has no encryption for its data files and the communication between the client and server or between its servers is performed without authentication and data encryption. Surprisingly, Hypertable has no vulnerabilities for the injection though it has an HQL for data management, which is similar to the SQL commands in the relational database. Additionally, there is no information affirming the denial of service attack launched on Hypertable reported in CVE [24] (cve.mitre.org).

E. *Redis*

Redis does not support the data encryption as similar as MongoDB and Cassandra, so hackers or anybody with an access to the Redis server will be able to read all data in the database.

There is no data encryption performed in the communication between Redis client and Redis server, and between other servers in either the same or different cluster, because Redis has been mainly designed to function rapidly, so it does not have many components for its security. For this reason, hackers with access to the network of Redis servers will be able to detect the data while it is being communicated.

The document on "Redis Security" [25] from Redis's official [26] website affirms that Redis does not have a concept of string escaping, so the injection becomes impossible. Besides, the researchers also looked for the CVE's vulnerabilities [24] from cve.mitre.org which is likely an archive of the security vulnerabilities from many computer systems, but neither injection nor denial of service attacks on Redis was reported.

IV. THE SECURITY ENHANCEMENT FOR THE TOP 5 OPEN SOURCE NoSQL DATABASES

A. *MongoDB*

The developers can protect MongoDB from hacking by encrypting the data in application level. That is, before writing any data, especially a sensitive one, e.g. password or credit card number, the application (e.g. web application) should perform the data encryption before recoding them in the data files.

Using the reverse proxy e.g. Apache HTTPD or mod proxy module will be helpful in the data encryption since all data will be allowed to run on the SSL and the authentication can be set up.

Running MongoDB in standalone mode or replica-set mode is more secured than in shared mode because the authentication with pre-shared secret is activated. However, hackers with an access to the system files can crack the pre-shard secret. Thus, to make the key file more secured, permission in an OS level should be determined to suit the key file (e.g. using chmod command).

To use the SQL injection and Cross Site Scripting (XSS) to attack the relational database, hackers need to insert some extra symbols into the system e.g. ('), ("), (<), and (>). In the same way, we can prevent this kind of attack on MongoDB by terminating the following symbols: (:), ({), and (}), in order to stop the attacking input from getting into the web server, which is the frontage of the database server. For these reasons, the developers should write an extra script to detect and delete these extra symbols before they can get into the database.

To prevent the denial of service attack, newer versions of MongoDB, further than version 2.4.4, should be installed.

B. Cassandra

To protect Cassandra from data hacking can be performed by encrypting the data in application level before being recorded into the data files. Additionally, an appropriately determined permission in an operating system can prevent data hacking from the unauthorized users.

We can switch the communication between the client and server of Cassandra to be the data encryption by enabling the SSL which is a component currently equipped with the new versions of Cassandra (version 0.8.xbrach or later).

To create the safer communication between nodes in a cluster or between clusters of Cassandra, the developers should use the new versions of Cassandra that supports data encryption (version stable 0.8 branch or later) together with using a private CA.

Although the CQL is secured, some report on its security vulnerabilities still be found. Still, the performance of input validation on the server e.g. preventing against a single quote (') and limit length of the input can prevent against the injection vulnerability that might be happen in the future.

To prevent the denial of service attack on Cassandra, the system should be designed to hide the server's IP address from the internet. At this point, the only problem that remains unsolved is the inactive connections.

C. CouchDB

There are two methods broadly used to protect CouchDB's data files from hacking viz., OS level disk encryption and application level encryption. The first method can be performed by managing the configuration in OS level which is an automatic data encryption with rapid function. The second method may functions much more slowly and demonstrates some limitation in searching and indexing the data.

To protect CouchDB from JSON injection can be performed by preventing against the attacking character (:), ({ }), and especially ({). In the document [20], it is suggested that the developer should make use of WAF (Web Application Firewall) to prevent against this kind of attack as well. From the experiment, the researchers used IPTABLES as WAF to detect the strings or patterns of the attacks. That is, the researchers used the "--m string" on IPTABLES and found that it could terminate JSON injection.

To protect CouchDB from the denial of service attack can be performed by 1.) upgrading the newest version of CouchDB, 2.) downloading and installing patch from the manufacturer's company to prevent the security vulnerability; and 3) following the instruction from the expert's websites regarding the security vulnerability such as www.securityfocus.com.

D. Hypertable

To enhance the security for Hypertable can be done in application level. That is, adding more codes in application to encrypt the sensitive data before being recorded in the database. However, Hypertable's official website does not suggest any method to secure the authentication and encryption for the data that is being communicated between the client and server or between the servers. It only says that Hypertable has been currently developed with a main focus on its functional speed, so the security component will be equipped in its upcoming version.

E. Redis

There are two methods to protect Redis's data files from hacking viz., the data encryption in application level and OS level encryption by determining a permission for the OS users so that only the privileged users can open and read the data files.

To prevent the data capturing during/sniffing the communication between the client and server, or between the servers and servers in either the same or different cluster can be performed by utilizing the SSL. In this regard, the researchers conducted an experiment with a tool called ‘stunnel’ [25] and found that it could encrypt the data very quickly and safely, since stunnel uses the SSL protocol with data compressing. The client and server were connected as illustrated in Fig 1.

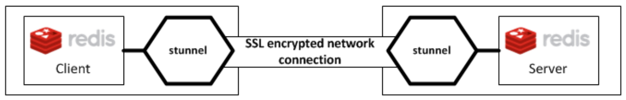

Fig 1. The Connection between Redis’s Client and Server using 'stunnel'

The client was configured to 127.0.0.1:36379 and it was supposed that Server IP = 192.168.1.99. The commands used in this experiment were as follows.

Server

```
stunnel -d 26379 -r 6379 -p /etc/ssl/redis.pem
```

Client

```
stunnel -c -v 3 -A /etc/ssl/redis.crt \
    -d 127.0.0.1:36379 -r 192.168.1.99:26379
```

Note: The researchers created the private key, related certificate, and self-signed certificate with the following commands.

```
openssl req -out /etc/ssl/redis.csr \
  -keyout /etc/ssl/redis.key \
  -nodes -newkey rsa:2048

openssl x509 -req -days 3650 \
    -signkey /etc/ssl/redis.key \
    -in /etc/ssl/redis.csr \
    -out /etc/ssl/redis.crt

cat /etc/ssl/redis.key /etc/ssl/redis.crt \
    >/etc/ssl/redis.pem
```

V. THE SECURITY COMPARISON OF THE TOP 5 OPEN SOURCE NoSQL DATABASES

According to the data in Section 3 and 4, it can be concluded as shown in Table 1. From Table 1, it was found that all of the five NoSQL databases did not support the data encryption but the first three databases (i.e. MongoDB, Cassandra and CouchDB) supported the authentication and encryption for the communication between the client and server and between the servers and servers. MongoDB and Cassandra were rated ‘weak’ because their features for authentication and encryption were disabled by default. Significantly, no statistical data mentions that Hypertable and Redis have been attacked by Script Injection or Denial of Service, but CouchDB still has problem with those kinds of attacks. Furthermore, it was indicated that MongoDB has a problem only with Script Injection as Cassandra server still encounters the Denial of Service attack.

Table 1. The Security Comparison of the Top 5 Open Source NoSQL Databases

Security Issues	Databases				
	MongoDB	Cassandra	CouchDB	Hypertable	Redis
Data files encryption	No encrypt	No encrypt	No encrypt	No encrypt	No encrypt
Client/Server Authentication /Encryption	weak	weak	SSL	No authen / No encrypt	No authen / No encrypt
Inter-cluster Authentication /Encryption	weak	weak	SSL	No authen / No encrypt	No authen / No encrypt
Script Injection	Vulnerable	Not vulnerable	Vulnerable	Not vulnerable	Not vulnerable
Denial of service attack	Not vulnerable	Vulnerable	Vulnerable	Not vulnerable	Not vulnerable

VI. CONCLUSION

This study has compared the security level of the top 5 open source NoSQL databases with a focus on 5 critical issues including: 1) Data file encryption, 2) Client/server authentication/encryption, 3) Inter-cluster authentication/encryption, 4) Script Injection, and 5) Denial of Service attack, aiming to offer the readers with a guideline for selecting the best database. In this regard, the researchers found that Hypertables and Redis are the most secured databases to handle the attack launched by internet users (mostly with Injection and DoS), and CouchDB, MongoDB, and Cassandra were the database that is mostly safe from data capturing/sniffing during the communication between the servers. Unfortunately, these five databases did not perform the data file encryption. In addition to the security comparison, the researchers suggested the useful methods to make these databases more secured, i.e., how to encrypt the sensitive data in the application level and to create the safer communication for the servers using stunnel.

REFERENCES

[1] James Manyika, “Big Data: The Next Frontier for Innovation, Competition, and Productivity,” Executive Summary, McKinsey Global Institute, May 2011, <http://www.mckinsey.com/mgi/publications/big_data/pdfs/MGI_big_data_exec_summary.pdf>.
[2] Zikopoulos, P., Eaton, C., deRoos, D., Deutsch, T., & Lapis, G. (2012). Understanding big data: Analytics for enterprise class hadoop and streaming data. New York, NY: McGraw-Hill.
[3] R. Cattell, "Scalable SQL and NoSQL Data Stores," ACM SIGMOD Record, vol. 39, December 2010.
[4] "NoSQL: An Overview of NoSQL Databases", Tim Perdue, http://newtech.about.com/od/databasemanagement/a/Nosql.htm, 2014
[5] "NoSQL", http://en.wikipedia.org/wiki/NoSQL, 2014
[6] "Oracle NoSQL Database", http://www.oracle.com/technetwork/database/database-technologies/nosqldb/overview/index.html, 2014
[7] "12 Best Free and Open Source NoSQL Databases", http://bigdataanalyticsnews.com/12-best-free-open-source-nosql-databases/, 2014
[8] " Five open-source NoSQL technologies worth looking at", https://econsultancy.com/blog/7407-five-open-source-nosql-technologies-worth-looking-at#i.108tt098v3f67z, 2011
[9] "Compare Open Source NoSQL Databases", http://nosql.findthebest.com/d/l/Open-Source, 2013
[10] "18 Best Open Source NOSQL Database Project To Build Highly Scalable Applications", http://www.fromdev.com/2012/07/best-open-source-nosql-database.html, 2012
[11] K. Chodorow and M. Dirolf, MongoDB: The Defitive Guide. Sebastopol, CA: O’Reilly and Associates, 2010.
[12] A. Malik and P. Lakshman, "Cassandra: a decentralized structured storage system," SIGOPS Operating System Review, vol. 44, no. 2, 2010.
[13] J. C. Anderson, J. Lehnardt, and N. Slater.CouchDB: The Definitive Guide. 2010.
[14] A. Khetrapal, and V. Ganesh. "HBase and Hypertable for large scale distributed storage systems" http://www.ankurkhetrapal.com/downloads/HypertableHBaseEval2.pdf,2006
[15] T. Macedo and F. Oliveira. Redis cookbook. O’Reilly Media, 2011.
[16] "Testing for NoSQL injection", https://www.owasp.org/index.php/Testing_for_NoSQL_injection, 2014
[17] "CVE Detail", http://www.cvedetails.com/vulnerability-list/vendor_id-12752/product_id-25450/version_id-147610/Mongodb-Mongodb-2.4.4.html
[18] "CQL injection attacks?", https://www.mail-archive.com/user@cassandra.apache.org/msg14829.html, 2014
[19] "Assessing Data Availability of Cassandra in the Presence of non-accurate Membership", L Aniello, S Bonomi, M Breno, R Baldoni, Proceedings of the 2nd International Workshop on Dependability Issues in Cloud Computing (DISCCO '13), http://www.dis.uniroma1.it/~midlab/articoli/discco2013.pdf
[20] " NoSQL, No Injection !?", Kuon, http://www.hitcon.org/hit2010/download/8_NoSQL_No_Injection.pdf, 2014
[21] "Apache CouchDB Universally Unique IDentifier (UUID) Remote Denial of Service Vulnerability", http://www.securityfocus.com/bid/66474/discuss, 2014
[22] "Exploit Title: Couchdb uuids DOS exploit # Google Dork", www.exploit-db.com/download/32519/, 2014
[23] "Bug 1082168 - (CVE-2014-2668) CVE-2014-2668 couchdb: remote denial of service flaw", https://bugzilla.redhat.com/show_bug.cgi?id=1082168, 2014
[24] "Common Vulnerabilities and Exposures", https://cve.mitre.org/, 2014
[25] "stunnel: Home", https://www.stunnel.org/index.html, 2014
[26] "Redis Security", http://redis.io/topics/security, 2014