International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

A Random Walk through the Dark Side of NoSQL Databases in Big Data Analytics

Kudakwashe Zvarevashe[1], Tatenda Trust Gotora[2]

[1]M Tech Student, Department of CSE, Jawaharlal Nehru Technological University, Hyderabad, India
500085, Kukatpally, Hyderabad, Andhra Pradesh, India

[2]M Tech Student, Department of Software Engineering, Jawaharlal Nehru Technological University, Hyderabad, India
500085, Kukatpally, Hyderabad, Andhra Pradesh, India

Abstract: ***The massive growth of technology has greatly stimulated the need to generate data. Every day people and companies generate enormous amounts of data and this data may be structured, unstructured, semi-structured or a combination of all. This has called for the need to design databases which can store this type and volume of data and NoSQL databases have been the antidote crafted and designed to alleviate this problem. However, the NoSQL solution to this Big Data problem has also lead to several other problems therefore this paper seeks to reveal this problem taking a walkthrough into the structure of NoSQL databases. In addition, the paper also makes a sneak preview of the discovered mainstream (MongoDB vs. Cassandra) NoSQL database security features analysis.***

Keywords: NoSQL, big data, structured data, unstructured data, semi-structured data, MongoDB, Cassandra**.**

1. Introduction

The manner at which today's data is scaling is getting more and more difficult to predict. Data is growing at an exponential rate such that it is believed that almost 90% of the world's data was generated over the last two years. Most business transactions are now being processed over the web, social networks have become the common man's factory of data where large amounts of data is being created, processed and stored. This data has become so big that traditional technologies like relational databases cannot handle its complexity as far as distributed storage and processing is concerned. This has since led to the introduction of NoSQL databases to the technological world.

In the years building up to 2004 [1] Google embarked on a project to set up their very own proprietary database which they named [3]"Big Table". Big Table was an instant hit and it solved many of the problems with relational databases. In 2006 Yahoo built the first prototype called Hadoop and in 2008 they went commercial. Amongst the companies that started implementing this technology, Facebook was the first to join followed by Twitter and the others.

The term NoSQL can easily mislead a lot of intellectuals because it sounds as if it means no SQL is allowed in this technology. In actual fact, NoSQL stands for [2]"Not Only SQL" which is an admission that the main aim here is not to reject SQL but rather to compensate for the technical limitations that are associated with relational databases.

2. Motivation

The continuous development of the Internet and cloud computing has stimulated a number of demands which have been big motivators behind NoSQL databases. Figure 1 below shows a list of these motivating factors.[4],[5]

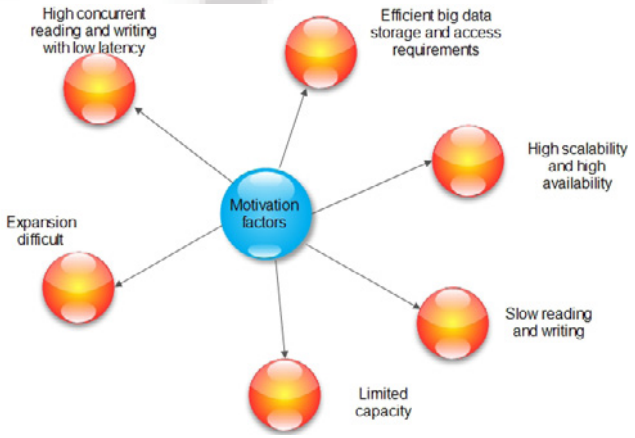

Figure1: Illustration of NoSQL Motivating Factors

- High Scalability and availability:
  Due to the increase in the number of concurrent requests and data, there is need for the databases to have strong support easy expansions without interrupting the services being offered.

- Slow reading and writing
  As the data increases relational databases become prone to deadlocks and concurrency issues and this has lead to a rapid decline in reading and writing.

- *Limited Capacity*
  The existing relational databases cannot support Big Data in search engines, SNS or Big System. The data needs of companies like Facebook, Google, Yahoo etc cannot be totally supported by relational databases.

- *Efficient Big Data storage and access requirements*
  Large applications such as SNS search engines need databases that meet the efficient storage (Petabytes level) and which properly responds to the needs of millions of

traffic.

- *Expansion Difficulty*
  Relational databases have poor scalability in terms of expansion due to multi-table correlation mechanism.

- *High Concurrency of reading and writing with low latency*
  Relational databases have low concurrencies with high latency which is the complete opposite of the required expectations in this data laden world. In order to satisfy the needs of customers, the database driven applications will have to respond quickly.

The following is a list of areas where NoSQL is currently being used.

- Social Networks
- Search engines
- Geospatial analysis
- Molecular modeling
- Data Warehousing
- Caching

3. The CAP Theorem

When designing a distributed system it is very important to understand the concept of CAP Theorem and NoSQL databases are no exceptions when it comes to these issues. In 2000, Professor Eric Brewer conjectured that a distributed system cannot simultaneously provide all three of the following desirable properties:

- **C**onsistency: A read sees all previously completed writes.
- **A**vailability: Reads and writes always succeed.
- **P**artition tolerance: Guaranteed properties are maintained even when network failures prevent some machines from communicating with others.

Therefore NoSQL databases can be classified using the CAP Theorem. The entire current NoSQL database follow the different combinations of the C, A, P from the CAP theorem. Here is the brief description of three combinations CA, CP, AP:

i)Consistency with Availability
These types of data bases are mainly concerned with consistency and availability. Systems concern the CA are: the traditional relational database, Vertica (Column-oriented), Aster Data *(*Relational) , Greenplum (Relational) and so on.

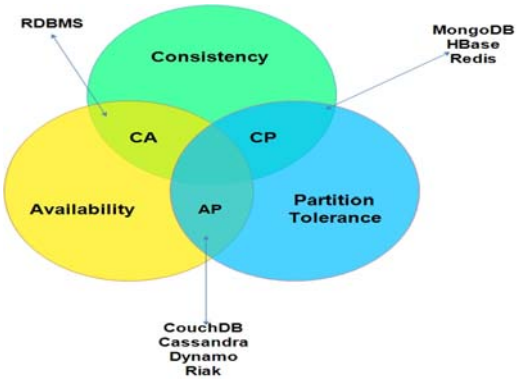

Figure2: Illustration of the CAP Theorem

ii)Consistency with Partition Tolerance
Such a database system stores data in the distributed nodes, but also ensures the consistency of the data, but it does not have good support for availability. Example of databases which employ the CP system are: BigTable (Column-oriented), Hypertable (Column-oriented), HBase(Column-oriented),MongoDB(Document),Terrastore(Document),Redis(Key-value),Scalaris(Key-value),MemcacheDB(Key-value), Berkeley DB (Key-value*)*.

iii)Availability with Partition Tolerance
Such systems ensure availability and partition tolerance primarily by achieving consistency, AP's system: Voldemort (Key-value), Tokyo Cabinet (Keyvalue), KAI (Key-value), CouchDB (Documentoriented), SimpleDB (Document-oriented), Riak*(*Document-oriented).

4. Top Challenges of NoSQL databases

The promise of the NoSQL database has generated a lot of enthusiasm, but there are many obstacles to overcome before they can appeal to mainstream enterprises [6]. The challenges of NoSQL databases can be grouped into a five letter achronym: BEAMS.

- Business Intelligence and Analytics
- Expertise
- Administration
- Maturity
- Support

i)Maturity
RDBMS systems have been around for a long time. NoSQL advocates will argue that their advancing age is a sign of their obsolescence, but for most CEOs, the maturity of the RDBMS is reassuring. For the most part, RDBMS systems are stable and richly functional. In comparison, most NoSQL alternatives are in pre-production versions with many key features yet to be implemented. Living on the technological leading edge is an exciting prospect for many developers, but enterprises should approach it with extreme caution [6].

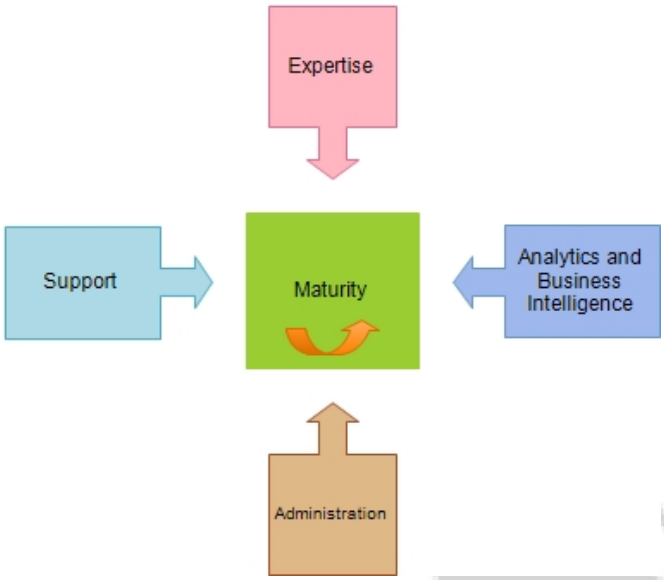

Figure 3: Top Challenges of NoSQL Databases

ii)Support
Enterprises want the reassurance that if a key system fails, they will be able to get timely and competent support. All RDBMS vendors go to great lengths to provide a high level of enterprise support.

In contrast, most NoSQL systems are open source projects, and although there are usually one or more firms offering support for each NoSQL database, these companies often are small start-ups without the global reach, support resources, or credibility of an Oracle, Microsoft, or IBM[6].

iii)Business Intelligence and Analytics
NoSQL databases have evolved to meet the scaling demands of modern Web 2.0 applications. Consequently, most of their feature set is oriented toward the demands of these applications. However, data in an application has value to the business that goes beyond the insert-read-update-delete cycle of a typical Web application. Businesses mine information in corporate databases to improve their efficiency and competitiveness, and business intelligence (BI) is a key IT issue for all medium to large companies.

NoSQL databases offer few facilities for ad-hoc query and analysis. Even a simple query requires significant programming expertise, and commonly used BI tools do not provide connectivity to NoSQL.

Some relief is provided by the emergence of solutions such as HIVE or PIG, which can provide easier access to data held in Hadoop clusters and perhaps eventually, other NoSQL databases. Quest Software has developed a product -- Toad for Cloud Databases -- that can provide ad-hoc query capabilities to a variety of NoSQL databases.

iv) Administration
The design goals for NoSQL may be to provide a zero-admin solution, but the current reality falls well short of that goal. NoSQL today requires a lot of skill to install and a lot of effort to maintain.

5. Data Security Challenges of NoSQL databases

Lior Ohkman et al in [7] reviewed two of the most popular NoSQL databases (Cassandra and MongoDB) and outlined their main security features and problems.

A. Overview of Cassandra and MongoDB

1) Cassandra

Cassandra is a database management system designed to handle very large amounts of data which is distributed across several servers while providing a highly available service with no single point of failure. Its characteristics are:

- The schema is very flexible and does not require the designing of a database schema first and then add and delete fields are very convenient.
- Supports range queries
- It has high scalability such that a single point of failure does not affect the whole cluster and it supports linear expansion.

The security features of Cassandra were summarized into a tabular format in Table1 below:

Table 1: Security Features in Cassandra

Category	Status	Recommendation
Data at rest	Unencrypted	Protect with OS Level mechanisms
Authentication	The available solution isn't production ready	
Authorisation	Done at the CF Granularity Level. The available solution isn't of production quality.	Implement a custom IAUTHORITY PROVIDER
Auditing	Not available OOTB.	Implement as part of the authentication and authorisation solutions
Intercluster Network communication	Encryption is available	Enable this using a private CA
Client Communication	No encryption is available	Add packet filter rules to prevent unkown hosts from connection. Re-implement the Thrift server-side to use the SSL transport in Thrift 0.6. Add timeouts for silent connections in the Thrift server-side and cap the number of acceptable client connections.
Injection Attacks	Possible in CQL	If using the java driver its better to use Prepared statements that to use Statements.

2) MongoDB

MongoDB (from"humongous") is a schema-free, document-oriented database written in the C++ programming language. The database is document-oriented in that it manages collections of schema-less JSON-like documents. This allows data to be nested in complex hierarchies and still be query-able and indexable[8]. Its features are:

- **Flexibility**
  MongoDB stores data in JSON documents (which we serialize to BSON). JSON provides a rich data model that seamlessly maps to native programming language types, and the dynamic schema makes it easier to evolve your data model than with a system with enforced schemas such as a RDBMS [8].
- **Power**
  MongoDB provides a lot of the features of a traditional RDBMS such as secondary indexes, dynamic queries, sorting, rich updates, upserts (update if document exists, insert if it doesn't), and easy aggregation. This gives you the breadth of functionality that you are used to from an RDBMS, with the flexibility and scaling capability that the non-relational model allows [8].
- **Speed/Scaling**
  By keeping related data together in documents, queries can be much faster than in a relational database where related data is separated into multiple tables and then needs to be joined later. MongoDB also makes it easy to scale out your database. Autosharding allows you to scale your cluster linearly by adding more machines. It is possible to increase capacity without any downtime, which is very important on the web when load can increase suddenly and bringing down the website for extended maintenance can cost your business large amounts of revenue [8].
- **Ease of use**
  MongoDB works hard to be very easy to install, configure, maintain, and use. To this end, MongoDB provides few configuration options, and instead tries to automatically do the "right thing" whenever possible [8]. This means that MongoDB works right out of the box, and you can dive right into developing your application, instead of spending a lot of time fine-tuning obscure database configurations. The security features of MongoDB were summarized into a tabular format in Table2 below:

Table1: Security Features in MongoDB

Category	Status	Recommendation
Data at rest	Unencrypted	Protect with OS Level mechanisms
Authentication for native connections	Available only in unsharded configurations	Enable if possible
Authorisation for native connections	READ/READ-WRITE/Admin levels only in unsharded connections	Enable if possible, requires enable authentication
Auditing	Not available in MongoDB.	Implement as part of the authentication and authorisation solutions
AAA (Authorisation,Authentication & Auditing) for RESTful connections.	Users and permissions are maintained externally	Available if configured on a reverse proxy
Database Communication	No encryption is available	
Injection Attacks	Possible in via Javascript or string concatenation.	Verify that the application does reasonable input validation.

6. Conclusion and Future work

In this paper we have managed to combine several problems discovered by several researches in NoSQL databases. We have also managed to reveal the security issues associated with NoSQL databases using a case study of Cassandra and MongoDB which was discovered by Lior Ohkman et al in [7]. The main problems they discovered common to both systems include lack of encryption support for the data files, weak authentication both between the client and the servers and between server members, very simple authorization without support for RBAC or fine-grained authorization, and vulnerability to SQL injection and Denial of Service attacks. For our future work we would like to design an authentication system that is production ready for cassandra an to design an API that counters the issues of Injections in NoSQL databases.

References

[1] Bill Vorhies, "A Brief History of Big Data Technologies from SQL to NoSQL to Hadoop and Beyond",para.4, Oct 31, 2013 http://data-magnum.com/a-brief-history-of-big-data-technologies-from-sql-to-nosql-to-hadoop-and-beyond/
[2] https://www.usenix.org/legacy/publications/login/2011-10/openpdfs/Burd.pdf
[3] Google BigTable: http://labs .google .com/papers/bigtable .html
[4] Kai Fan, "Survey on NoSQL", Programmer, 2010(6): pp 76-78
[5] Jing Han, Meina Song, and Junde Song, A Novel Solution of Distributed Memory NoSQL Database for Cloud Computing", in ICIS 2011, 10th IEEE/ACIS International Conference on Computer and Information Science, 2011
[6] Guy Harrison "Ten Things You Should Know About NoSQL Databases", para.7, Aug 26, 2010 http://www.techrepublic.com/blog/10-things/10-things-you-should-know-about-nosql-databases/
[7] Lior Okman, Nurit Gal-Oz, Yaron Gonen, Ehud Gudes, Jenny Abramov, "Security Issues in NoSQL Databases" 2011 International Joint Conference IEEE.
[8] http://www.mongodb.org/about/introduction/

Author Profile

Kudakwashe Zvarevashe: Attained his BSc degree in Information Systems at MSU, Zimbabwe in 2010. He is currently doing M Tech IT final year at JNTUH, India. He is a HIT staff development research fellow. His research interests are in the area of big data, information security, cloud computing and web services.

Tatenda Trust Gotora: Attained his BSc degree in Computer Science at MSU, Zimbabwe in 2011. He is currently doing M Tech SE final year at JNTUH, India. He is a HIT staff development research fellow. His research interests are in the area of mobile computing, big data, information security and web services.