I J C T A, 9(28) 2016, pp. 181-189

SURVEY ON PROMINENT PRIVACY PRESERVING TECHNIQUES USED FOR PROVIDING SECURITY TO BIG DATA

D. Anuradha* and S. Bhuvaneswari**

Abstract: We are facing with a torrent of data generated and captured in digital form as a result of the advancement of sciences, engineering and technologies, and various social, economical and human activities. Big data applications provide a great benefit to many large scale and small scale industries. Big Data creates critical information security and privacy problems, at the same time Big Data analytics promises significant opportunities for prevention and detection of advanced cyber-attacks using correlated internal and external security data. We must address several privacy and security challenges to realize true potential of Big Data for information security. The paper analyzes Big Data applications for information security problems, and defines research directions on Big Data analytics for security intelligence. Traditional encryption solutions can protect the data but they can't be used to compute on encrypt data, however a novel encryption scheme, called fully homomorphic encryption (FHE), could compute over encrypted data without decrypting it. Extracting valuable information from attributes by evaluating them is the main goal of analyzing big data which need to be protected. Since, it is impossible to protect all big data, we consider big data as a single object which has its own attributes, and the set of attribute which have a higher relevance is more important than other attributes. A novel keyword search method to enable customers easily searching keywords from encryption-protection data is also discussed in this paper.

Keywords: Homomorphic encryption, Attribute evaluation, keyword search method

INTRODUCTION [13]

Big data can be described as huge volume (Zetta bytes) of different types of data. The next important aspect is that the growing rate of size of the Big data. This makes the Big data management a challenging task. As the name implies the Big data is not only big, but also possesses other important aspects. There are different ways in which the Big data is defined. The most appropriate one describes the three important features called **3V:** Volume, Velocity and Variety. Since the trueness of the data being processed is very important, IBM has introduced the fourth feature called 'Veracity'. Later Oracle has added the fifth aspect called the 'Value', which is represents the importance of Big data analytics. The following describes the properties of Big data:

Handling huge amount of data efficiently for arriving at a decision is called Big data management. The exact definition of Big data can be given using its properties. [2]

(i) **Volume:** The amount of data is characterized by volume.

(ii) **Velocity:** It represents the speed of data coming from various sources.

(iii) **Variety:** Different categories of data like traditional, structured , semi structured and unstructured data from web pages, sensors, social media, etc. are be handled in Big data.

(iv) **Variability:** It refers to the inconsistency of the data flow.

(v) **Value:** Efficient handling and filtering of data for a query adds value to the business.

* Research scholar (PT), Department of Computer Science, Pondicherry University. **Email:** anuradha.d@vit.ac.in

** Prof & HOD, Department of Computer Science, Pondicherry University. **Email:** booni_67@yahoo.co.in

(vi) **Complexity:** It measures the difficulties in linking, matching, transforming, correlating relationships and hierarchies of the data coming from various sources.

1.1 Issues and challenges of Big data [1]

- ✓ **Privacy and security:** When personal data are to be combined with large data set, new inferences about that person can be done by the data owner. In order to analyze and to take a decision the information about users are to be collected and stored. This may not be known to the users. Literate people may take advantage of Big data analysis and decision making, whereas under privileged cannot do that.
- ✓ **Data access and sharing information:** Since Big data is used for making decision on accurate results in time, it is necessary to make the data available in accurate, complete and in time. Also sharing of data in time may decrease the degree of completeness and accuracy.
- ✓ **Storage and processing issues:** The classical storage used is not enough for Big data. Uploading this large amount of data in **cloud** is not feasible, since it will take more time and data will grow rapidly. At the same time analysis requires complete data. Hence these **cloud issues** with Big data are categorized into capacity and performance issues.

  Transportation of Big data over the net is also cumbersome. Two ways to avoid transportation only the n are 1) processing the Big data at the storage itself and 2) transport only the data need to be processed, instead of complete data. Even processing of these data takes huge time.
- ✓ **Analytical challenges:** We need to formulate the analyzing procedure, if the volume and variety of data rapidly increases. Decision has to be made whether all the incoming data have to be stored or not and whether all the stored data need to be analyzed. Finding the important part of Big data for analysis and how to get best advantage of Big data are the few challenges ahead in this field.
- ✓ **Skill requirement:** Since it is an emerging technology, it is not employed in most of the industries and many people are not aware of it. So the the required number of experienced and skilled people are not available to solve the difficulties in Big data field.
- ✓ **Technical challenges:** Complete fault tolerance is not possible in an feasible way. So the main task is to reduce the failure to an 'acceptable' level. Scalability issue of Big data has lead towards **cloud computing**, since high level of sharing of resources with least expense is required.

  Big data basically focuses on quality data rather than having very large irrelevant data. Analyzing and mining of heterogeneous data is a challenging task for the solution developers.

2. CLASSIFICATION OF PRIVACY PRESERVING METHODS

2.1 Encryption [15]

The primary purpose of encryption is to protect the confidentiality of digital data stored on computer systems or transmitted via the Internet or other computer networks. Modern encryption algorithms play a vital role in the security assurance of IT systems and communications as they can provide not only confidentiality, but also the following key elements of security:

- *Authentication:* the origin of a message can be verified.
- *Integrity:* proof that the contents of a message have not been changed since it was sent.
- *Non-repudiation:* the sender of a message cannot deny sending the message.

Data, often referred to as plaintext, is encrypted using an encryption algorithm and an encryption key. This process generates cipher text that can only be viewed in its original form if decrypted with the correct key. Decryption is simply the inverse of encryption, following the same steps but reversing the order in which the keys are applied.

2.2 Public auditing [3]

To fully ensure the data integrity and save the cloud users' computation resources as well as online burden, it is of critical importance to enable public auditing service for cloud data storage, so that users may resort to an independent third-party auditor (TPA) to audit the outsourced data when needed. The figure 1 depicts the working of third-party auditing. The TPA, who has expertise and capabilities that users do not, can periodically check the integrity of all the data stored in the cloud on behalf of the users, which provides a much more easier and affordable way for the users to ensure their storage correctness in the cloud.

Moreover, in addition to help users to evaluate the risk of their subscribed cloud data services, the audit result from TPA would also be beneficial for the cloud service providers to improve their cloud-based service platform, and even serve for independent arbitration purposes. In a word, enabling public auditing services will play an important role for this nascent cloud economy to become fully established; where users will need ways to assess risk and gain trust in the cloud. Public audit ability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data

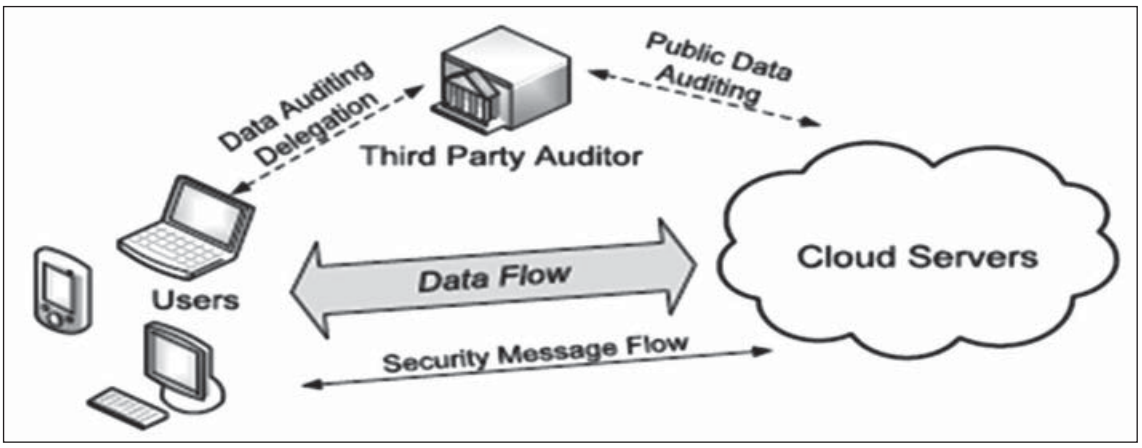

Fig1: Architecture of Cloud storage with TPA

A public auditing scheme consists of four algorithms (KeyGen, SigGen, GenProof, VerifyProof). KeyGen is a key generation algorithm that is run by the user to setup the scheme. SigGen is used by the user to generate verification metadata, which may consist of digital signatures. GenProof is run by the cloud server to generate a proof of data storage correctness, while VerifyProof is run by the TPA to audit the proof. Running a public auditing system consists of two phases, Setup and Audit:

- **Setup:** The user initializes the public and secret parameters of the system by executing KeyGen, and preprocesses the data file F by using SigGen to generate the verification metadata. The user then stores the data file F and the verification metadata at the cloud server, and deletes its local copy. As part of preprocessing, the user may alter the data file F by expanding it or including additional metadata to be stored at server.
- **Audit:** The TPA issues an audit message or challenge to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message by executing GenProof using F and its verification metadata as inputs. The TPA then verifies the response via VerifyProof.

Generally, the TPA is stateless, i.e., TPA does not need to maintain and update state between audits, which is a desirable property especially in the public auditing system. Note that it is easy to extend the framework above to capture a state full auditing system, essentially by splitting the verification metadata into two parts which are stored by the TPA and the cloud server, respectively. If the user wants to have more error resilience, he can first redundantly encodes the data file and then uses our system with the data that has error correcting codes integrated.

2.3 Access control [16]

Access control is a security technique that can be used to regulate who or what can view or use resources in a computing environment. There are two main types of access control: physical and logical. Physical access control limits access to campuses, buildings, rooms and physical IT assets. Logical access limits connections to computer networks, system files and data.

The four main categories of access control that are implemented in big data environment are Mandatory access control, Discretionary access control, Role-based access control, and Rule-based access control. Access control systems perform authorization identification, authentication, access approval, and accountability of entities through login credentials including passwords, personal identification numbers (PINs), biometric scans, and physical or electronic keys.

2.4 Authentication [12],[14]

Authentication is the process of determining whether someone or something is, in fact, who or what it is declared to be. Logically, authentication precedes authorization (although they may often seem to be combined). The two terms are often used synonymously but they are two different processes.

Authentication is a process in which the credentials provided are compared to those on file in a database of authorized users' information on a local operating system or within an authentication server. If the credentials match, the process is completed and the user is granted authorization for access. The permissions and folders returned define both the environment the user sees and the way he can interact with it, including hours of access and other rights such as the amount of allocated storage space. The process of an administrator granting rights and the process of checking user account permissions for access to resources are both referred to as authorization. The privileges and preferences granted for the authorized account depend on the user's permissions, which are either stored locally or on the authentication server. The settings defined for all these environment variables are set by an administrator

3. SURVEYED RESEARCH AREAS

3.1 Homomorphic Encryption

A Homomorphic Encryption (HE) scheme encrypts data in such a way that computations can be performed on the encrypted data without knowing the secret key. It is the encryption method with the property such that performing a function on two values separately and then encrypting the result yields the same final value as first encrypting two values separately and then applying the function to the results. So, given two encryptions $c_1 = E_{pk}(m_1)$ and $c_1 = E_{pk}(m_2)$ of messages m_1 and m_2 under public key pk, a HE scheme allows anyone to compute an encryption $E_{pk}\ m_1 \otimes m_2$ without needing to decrypt either c_1 or c_2. Here $\otimes$ denotes some arbitrary operation.

3.1.1 Partially Homomorphic Encryption algorithms [7], [8]

These are the homomorphic algorithms, they support only few operations. They are homomorphic over a small set of operations. For example, RSA encryption and Elgamal algorithms are multiplicatively

homomorphic and Elgamal algorithm is homomorphic with exponentiation operation also, but with constant only. Goldwasser-Milcali algorithm is homomorphic with XOR operation only. Benaloh, Naccahe-Stern and Paillier algorithms are additively homomorphic. Naccahe-Stern and Paillier algorithms allow homomorphic multiplication with constants.

3.1.2 Somewhat Homomorphic Encryption Algorithms [7], [8]

These homomorphic algorithms allow the functions to be repeated for limited number of times. This limitation is created by the error value term, which is generated in each operation. This value keeps increasing with each operation. When the error value exceeds certain value, the result no longer is decrypted. For example, Boneh-Goh-Nissim algorithm supports unlimited additions and single multiplication. The function modules for the above two schemes are KeyGen, Encrypt and Decrypt

3.1.3 Bootstrapping [7], [8]

It is the process of converting somewhat homomorphic encryption scheme into fully homomorphic encryption scheme. A somewhat homomorphic cryptosystem is bootstrappable if it can evaluate its own decryption with a sufficiently small error constant. This enables the implementation of a Re-crypt function which homomorphically decrypts the message and re-encrypts, reducing the error constant. The function modules used are KeyGen, Encrypt, Evaluate and Decrypt

3.1.4 Fully Homomorphic algorithms [7], [8]

If a homomorphic encryption scheme support two specific operations, namely, addition and multiplication, then it can support any operation; and these are called Fully homomorphic encryption algorithms. In 2009, the first fully homomorphic encryption algorithm was developed by Craig Gentry at Stanford. He brings in the bootstrap algorithm, which recrypts the plain text with another key, without decryption. In order to make the decryption correct, bootstrapping is done before every evaluation. Later in 2010 the same was revised by van Dijk, Gentry, Halevi, and Vaikuntanathan for integers. They implemented the algorithm using NAND gates. Any logical circuit can be constructed using NAND gates alone. In general, the procedure of these algorithms can be depicted diagrammatically as in figure 2.

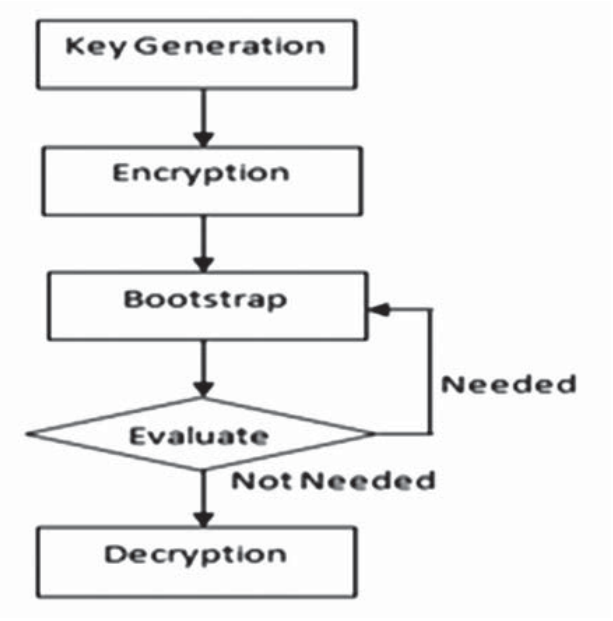

Figure 2. Fully homomorphic encryption

3.1.5 Fully Homomorphic Encryption in Big data [9]

Encryption scheme is additive and multiplicative homomorphic. In this we consider two big prime numbers A and B. The computation is given below:

$$P = A*B$$

Assuming a random positive integer Ar and message M, the encryption M is as follows.

$$\text{Cipher text } C = (M+A*Ar) \bmod P$$

$$\text{Cipher text C is decrypted as } M = C \bmod P$$

Assume two cipher texts of the form,

$$C1 = (M1+A*Ar1) \bmod P \text{ and}$$

$$C2 = (M2+A*Ar2) \bmod P$$

The addition operation will be performed as given below

$$C1+C2 = (M1+A*Ar1) \bmod P+(M2+A*Ar2) \bmod P$$

$$= ((M1+M2) + A*(Ar1+Ar2)) \bmod P$$

The multiplication operation is done as follows.

$$C1*C2 = (M1+A*Ar1) \bmod P * (M2+A*Ar2) \bmod P$$

$$= ((M1*M2) + A*((M1*Ar2)+(Ar1*M2)+(A*Ar1*Ar1)) \bmod P$$

The Reduce function finds the same keys and merges them. Since the system produce different keys for same value for the robustness of the security. So, the following **Tranform** function is applied on the keys.

For the cipher text C,

$$\mathbf{C^* = C*B*R \bmod P}, \text{ where R is a random positive integer}$$

Proof: For two cipher texts C1 and C2,

$$C1^* = C1*B*R \bmod P, \quad C2^*=C2*B*R \bmod P \text{ and since } P=A*B,$$

$$C1^*-C2^* = (C1*B*R - C2*B*R) \bmod P$$

$$= ((C1- C2)*B*R) \bmod P$$

$$= (((M1-M2) + A*(Ar1-Ar2)) *B*R) \bmod P$$

$$= (M1-M2) *B*R \text{ , since P is very large}$$

So, if $C1^* = C2^*$, then M1 = M2 otherwise $M1 \neq M2$

3.2 Attribute relationship evaluation [10]

The information of the attributes of Big data are essential for Big data analysis. All the attributes have some relationship with other attributes of different data sets. To provide security to the data we need to first select the valuable attributes to be made safe. This is done using attribute evaluation methods. Attribute evaluation is done to the data sets depending upon the characteristics of the data sets and requirements of the data processing. Here are few evaluation methods to be analysed. In this method each relation is assigned with a weight for comparison and evaluation, depending upon the data processing.

- Evaluation 1 [5]

✓ Equivalence relation

Attribute-*x* of data set X = Attirbute-y of data set Y

✓ Hierarchical relation

Attribute-xof data set X is not equal to Attirbute-*y* of data set Y

and Attribute-*x* is a subset of Attirbute-*y* or Attribute-*y* is a subset of Attirbute-*x*

✓ Unknown

There is no relationship between Attribute-x of data set X and attribute-*y* of data set Y

- Evaluation 2 [4]

Consider a counter for each attribute that counts the number of relationships an attribute has with other attribute. Increment the counter of each attribute for each relationship, the relation involves. With the help of this count we can decide that the attribute that has the highest count is the important attribute.

3.3 Type based keyword search [6]

In this scheme data files are appended with the key words. These key words are useful in identifying the file while search. The files are encrypted along with their key words. This search is called the public key encryption with keyword search (PKES). The data m and the extracted keywords w_1, w_2, w_3, …, w_n are encrypted using a public key pk and organized as follows;

$$\text{PKE}(pk, m) \,||\text{PKES}(pk, w_1)||\text{PKES}(pk, w_2)||\text{PKES}(pk,w_3)||\ldots||\text{PKES}(pk, w_n)$$

User create a trapdoor with a keyword w_i. The data server do search on the encrypted files and returns all the files that contains w_i. With the PKES system, when a trapdoor of key word w_i is submitted all the files containing the key word w_i are returned for decryption. The search can be improved by considering the type of the file also. In Type based PKES the key words are encrypted using the type also. In this scheme the data files are encrypted with respect to their type. Searching is done with the encrypted search keys and their type. First only the keywords are decrypted and then data files with the matching keywords alone are decrypted for result.

3.4 Map reduce Paradigm[11]

MapReduce is a most appropriate processing technique and a program model for Big data computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job.

The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model.

3.5 The Algorithm

Generally MapReduce paradigm is based on sending the computer to where the data resides. MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage. The figure 3 explains mapping and reducing jobs diagrammatically.

- **Map stage** : The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
- **Reduce stage** : This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster. The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes. Most of the computing takes place on nodes with data on local disks that reduces the network traffic. After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

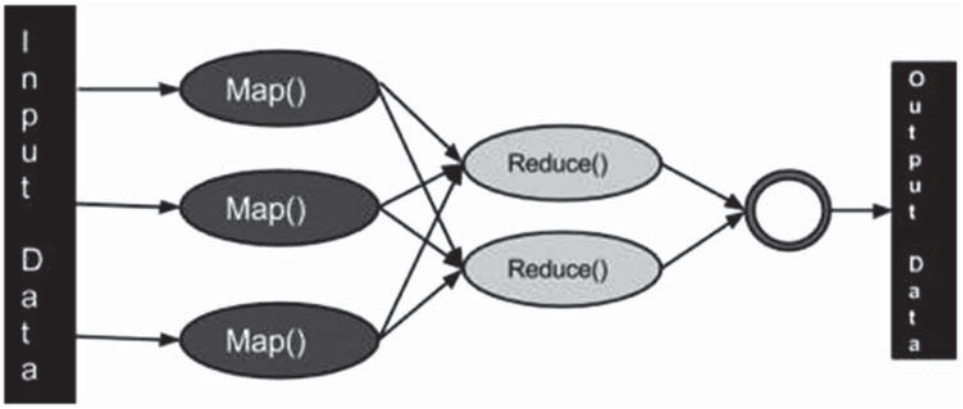

Figure 3. Inputs and Outputs (Java Perspective)

The MapReduce framework operates on <key, value> pairs, that is, the framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job, conceivably of different types. The key and the value classes should be in serialized manner by the framework and hence, need to implement the Writable interface. Additionally, the key classes have to implement the Writable-Comparable interface to facilitate sorting by the framework.

Input and Output types of a MapReduce job: (Input) <k1, v1> → map → <k2, v2> → reduce -> <k3, v3>(Output).

	Input	*Output*
Map	<k1, v1>	list (<k2, v2>)
Reduce	<k2, list(v2)>	list (<k3, v3>)

The output of the reduce job produces a file consists of key-value pairs of different types.

4. CONCLUSION

The first surveyed topic is homomorphic encryption scheme. It is a powerful and yet a simple encryption method especially for Big data. The classic homomorphic encryption scheme is augmented so as to be compatible for Big data. Since the volume of the Big data is the main challenge to be addressed, the

encryption should be simple but should be powerful for providing privacy. Encrypting the entire volume of Big data is not an efficient privacy preserving technique. The attribute evaluation method selects a set of attributes for encryption by evaluating the attributes. While encrypting the Big data, the type of the selected attributes can also be considered for improved performance.

5. FUTURE WORK

The existing homomorphic encryption can be simplified by simple arithmetic operations. This will reduce the encryption and decryption time. In the SQL data base each attribute can be encrypted with respect to their type. The queries on the data base can be easily solved by decrypting only the attributes required. Attribute evaluation method can be used for identifying the required attributes depending on the system requirements.

References

1. Marisa Paryasto, Andry Alamsyah, Budi Rahardjo, Kuspriyanto, "Big-Data Security Management Issues", 2014 2nd International Conference on Information and Communication Technology (ICoICT)
2. K. Davis and D. GordonPatterson. "Ethics of Big Data. O'Reilly", 2012.
3. Wang Shao-huiP, Chang Su-qinP, Chen Dan-weiP, Wang Zhi-weiP, "Public Auditing for Ensuring Cloud Data Storage Security With Zero Knowledge Privacy",
4. Sung-Hwan Kim, Jung-Ho Eom, Tai-Myoung Chung, "Big data Security Hardening Methodology using Attributes Relationship", 978-1-4799-0604-8/13, 2013 IEEE
5. Sung-Hwan Kim, Nam-Uk Kim, Tai-Myoung Chung, "Attribute Relationship Evaluation Methodology for Big Data Security", 978-1-4799-2845-3/13 , 2013 IEEE
6. Yang Yang, Xianghan Zheng, "Type based Keyword Search for Securing Big Data", International Conference on Cloud Computing and Big Data, 2013
7. c. Gentry, A fullyhomomorphic encryption scheme,Ph.D.dissertation, Stanford University, 2009
8. MY Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan,Fully Homomorphic Encryption over the Tntegers.;Tn Proceedings of EUROCRYPT. 2010, 24-43.
9. Xu Chen, Qiming Huang, "The Data Protection of MapReduce Using Homomorphic Encryption", 978-1-4673-5000-6/13, 20 13 IEEE
10. E.F. Codd, "A relational model of data for large shared data banks" ,Commun ACM, vol. 13, no. 6, pp377-387 1970.
11. Dean, J., & Ghemawat, S., "MapReduce: Simplified Data Processing on Large Clusters," Commun ACM 51, 2008, pp. 107-113.
12. Venkata Narasimha Inukollu, Sailaja Arsi, Srinivasa Rao Ravuri, SECURITY ISSUES ASSOCIATED WITH BIG DATA IN CLOUD COMPUTING" International Journal of Network Security & Its Applications (IJNSA), Vol.6, No.3, May 2014
13. A, Katal, Wazid M, and Goudar R.H. "Big data: Issues, challenges, tools and Good practices.". Noida, 2013, pp. 404 – 409, 8-10 Aug. 2013.
14. M. Einar, N. Maithili, T. Gene. "Authentication and integrity in outsourced databases", ACM Transactions On Storage. 2006, 2(2): 107-138.
15. Ganugula U., Saxena A., "High Performance Cryptography: Need of the Hour," CSI Communications, pp. 16-17, September 2013.
16. S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing", in *INFOCOM'10*. IEEE, 2010, pp. 534–542.