Combined approach of Tokenization and Mining to secure and optimize Big Data in Cloud Storage

Shanto Roy*, Ahmedur Rahman Shovon† and Dr. Md. Whaiduzzaman ‡
Institute of Information Technology*†‡
Jahangirnagar University, Dhaka-1342, Bangladesh
Email: sroy.iitju*@gmail.com, shovon.sylhet†@gmail.com, wzaman‡@juniv.edu

***Abstract*—The era of technology is now shifting towards the Cloud Computing. After the term Big Data has arisen and computation tends to be provisioned as a service rather than a product. Recently Cloud Computing has become more portable and flexible in such way so that we call it having a super computer in our pockets. Despite the potential application of cloud computing, data security is still questionable in privacy issue due to the lack of location transparency. In Big Data arena users are confused whether their confidential data is being stored or being used for evil purposes. In this paper, a detail analysis of data privacy and security issues are discussed as well as an enhanced framework of security model is proposed with a view to eradicating the privacy issue. Tokenization provide a wider range of data security by protecting data from malicious insider threats or data breaches as well as perform storage optimization in the Big Data cloud with a little prior mining to convert Big Data into really small ones.**

I. Introduction

Cloud computing is the delivery of computing as a service rather than a product which combines shared hardware and software resources and deliver optimized enhancement and utilization of computing towards users. Cloud utilizes the computing resources along with reducing the cost with usage based options which makes it valuable towards the users. Now, there are a few quintillion bytes of data created every day including sensitive information that could mean profits for cyber-criminals and a single data breach may cost upto $1 billion. The more data a project requires, the bigger the threat to big data security and privacy. So, with Variety, Volume and Velocity; Big Data need to be protected in a scalable and efficient way. This paper describes a privacy preserving architecture that tokenizes big data and map towards distributed storage nodes. The system focuses on a primary master node which receives data, mine the data for a particular timestamp, tokenize the mined data using token vaults and then save tokenized data in storage nodes. Also, if any application wants to use the data, it requests the master node and a secure gateway is established for the application that accesses detokenized data in master node. The stored tokenized data in storage nodes does not need additional encryption methods which may be vulnerable during cryptanalysis or key management systems so that the system has an enhanced performance. Also, the tokenized data can't be detokenized unless someone has access to the token vault which ensures privacy of data against malicious insider threat or data breaches. A conceptual diagram of the proposed system is illustrated in Figure 1.

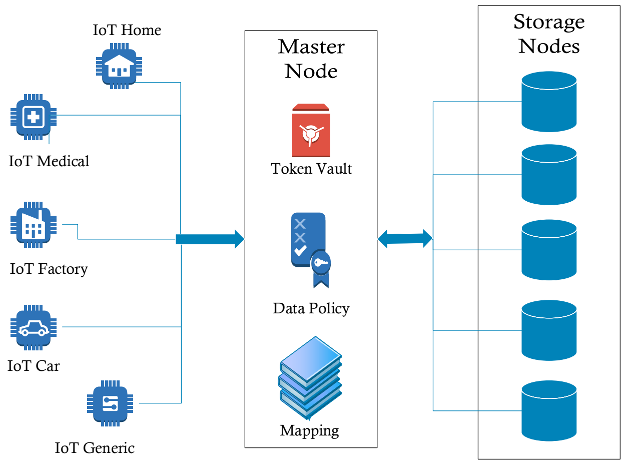


Fig. 1. Block Diagram of the proposed system

The primary contributions of this work are as follows:

- A secure architecture towards data privacy in cloud
- Conversion of Big data to small data with a little prior mining (for sensor readings)
- Tokenization process of data

II. Background Overview

There are already many prototypes and infrastructures for securing big data inside the cloud environment. Maximum works based on the security algorithms and ciphering/deciphering process are applied for securing the network channel or the storage data. In many works, some additional layers are added to the system to ensure confidentiality, integrity and availability of data. And very recently, the distributed storage mapping has been proposed as a good solution towards data privacy as insider threat arises the question that if the cloud is really secure whatever it is a public or private cloud.

In [1], authors addressed different cloud security issues such as trusting the third party cloud provider and different security identification of threats based on the CIA triad (Confidentiality, Integrity and Availability). The article assessed that public cloud can be relied upon based on low and high level confidentiality, server and client Authentication, creation of security domains, cryptographic separation of data and certificate-based authorization processes. Another article reviewed threats, vulnerabilities, security issues and their countermeasures based on the service delivery SPI model (SaaS, PaaS, IaaS) in [2]. Some of the countermeasures are as follows: identity and access management, dynamic credentials for preventing account or service hijacking; digital signatures, homomorphic encryption or AES in storage for data leakage; web application scanners for customer data manipulation, HyperSafe, trusted virtual Datacenter for VM escape etc. [3] also discussed security issues on basis of the SPI model. Authors focused primarily on the SaaS security issues such as security, locality, integrity, segregation, confidentiality, access and breaches of data; network security, authentication and authorization, web application security, virtualization vulnerability, availability, backup, identity management and sign-on process etc. As a solution towards the problems they mentioned about ”Cloud Security Alliance (CSA)security best practices for cloud computing,” 2009 [4] where CSA documented the future best practices to secure the cloud.

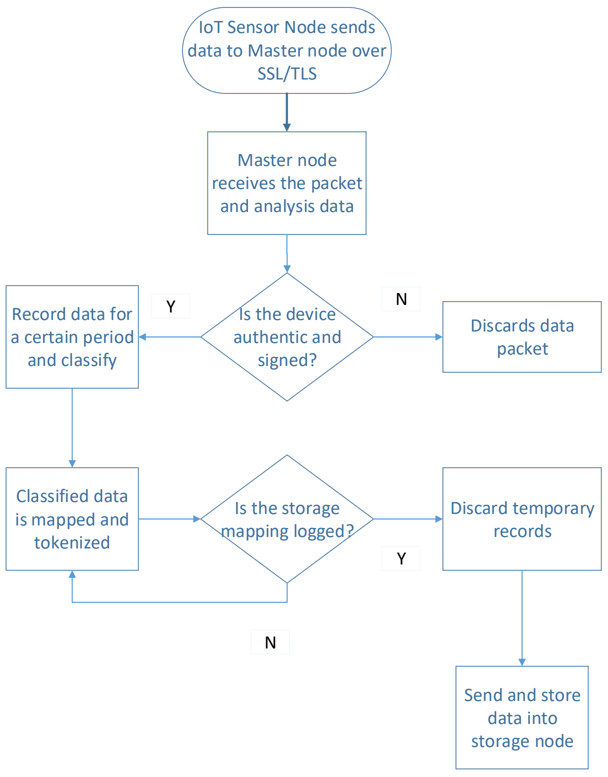
[7] discussed about current challenges and future research areas related to the privacy and security of big data such as privacy-preserving social network mining, privacy-preserving big data analytics, security aspects of big data exchange etc. Sophia Yakoubov et al. discussed about the current cryptographic approaches for the big data analytics and differentiated among Homomorphic encryption (HE), Verifiable computation (VC) and Multi-Party Computation (MPC) in [5]. Authors showed how HE and VC together can ensure the confidentiality and integrity of data, but together both encryption methods affect the performance. Thus MPC can be a good choice for current big data analytics where data will be distributed to multiple third party nodes so that attacker won’t be able to get the complete data but only a portion. In [6], authors tried to introduce an architecture of Smart-Frame, flexible, scalable and secure information management framework based on cloud computing. They designed the framework at three hierarchical levels: top, regional and end-user levels in which the first two levels are based on the cloud computing environment while the last level means the end-user smart devices. They also introduced identity-based cryptographic schemes, identity-based signature and identity-based proxy re-encryption to secure the information flow between the layers. Whereas a better modified identity-based proxy re-encryption model is proposed in [8]

[9] proposed an approach named Silverline towards data confidentiality in cloud with the help of another organization which will manage decryption keys and the encrypted data will be stored in storages. The client application fetch the two and decrypt the data locally to obtain the plain data. The approach is good as it removes the possibility of insider threat but another organization must be a trusted one so that keys are not compromised. Also it can affect overall performance in terms of Big data. In another work, [10] suggests PasS (Privacy as a Service); a new set of security protocols for ensuring data privacy in cloud environment with the help of secure computing through the use of cryptographic co-processors. Authors claim that the co-processor sharing mechanism employed in PasS will split the cost of the privacy services between the cloud provider and the customers. And finally CSA and ENISA continues to research on the best practices to secure Big Data in cloud [11] [12].

So, in order to preserve the privacy within cloud storage environment, most of the recent works discuss about encryption methods of data which are prone to cryptanalysis. Public clouds e.g. Google Cloud is using AES to encrypt data in storage [13] whereas Microsoft Azure is using both AES and bitlocker [14]. But, tokenization of data doesn’t need additional encryption inside the storage node which results in better performance and least cost as well as eradicates the possibility of insider threat or data leakage.

III. Methodology

The architecture is designed primarily to tokenize incoming sensor reading received by the master node and then map tokenized value towards distributed storage nodes. The system flowchart is shown in Figure 2.



IoT Sensor Node sends data to Master node over SSL/TLS
Master node receives the packet and analysis data
Is the device authentic and signed?
Y
N
Record data for a certain period and classify
Discards data packet
Classified data is mapped and tokenized
Is the storage mapping logged?
Y
N
Discard temporary records
Send and store data into storage node

Fig. 2. System flowchart

A. System design

1) Master node: Master node is the most important node in this work as it is the primary control panel for receiving sensor node data, tokenizing the data and mapping towards storage. Also it's responsible for controlling the data migration and granting the applications a secure access towards data for further mining. Figure 3 shows the elements and functionalities of the master node. Data analysis, review and mapping processes are performed inside the master node. Token vaults are used for tokenizing and detokenizing data. Like Hadoop file system (HDFS), master node can have job tracker and task tracker that helps mapping of data towards the storage nodes; and unlike HDFS, it can provide storage data security as well because it doesn't need to think about mapping plain data but only tokenized data.

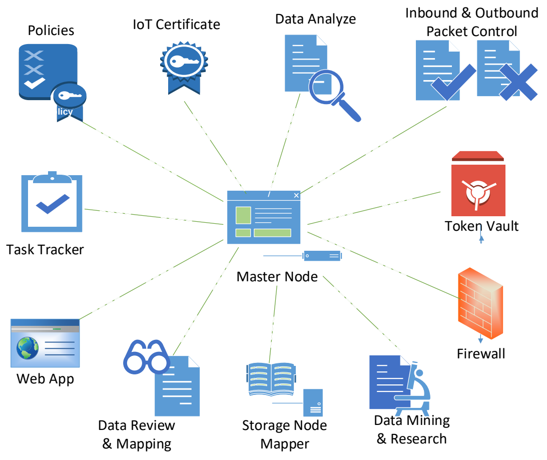


Fig. 3. Elements and Functionalities of Master node

2) Storage nodes: For secure storage management data can be mapped among distributed storage nodes. As data is tokenized before entering the storage nodes, data owner doesn't need to worry about the privacy of data. So, master node can use one or more third party storage nodes to store the data; although it's quite better to keep the data in distributed nodes to ensure the availability of data by keeping record of same data in minimum two nodes.

3) Policies: Policies include a clear definition pertaining cloud, its associated services, architectures, security certification and standards, clear understanding on the implications, rich rules in firewall, IoT certificates etc. Policies should be transparent, flexible and compatible with the existing systems.

4) Token Vaults: Token vaults are primarily designed to tokenize data value of different range including the analyzed or prior mined data in such a way that it is very lightweight and can be stored directly in the storage nodes without converting them into plain texts. Also, provides a secure gateway for the applications to access plain data after detokenization.

5) Storage Node Mapping: Every tokenized data packet is stored in minimum two or three different nodes. For tokenization minimum one token vault is needed to perform tokenization which is shown in Figure 4. Although it's better to have single token vault per storage as it can be precious for two particular reasons: optimization and random different

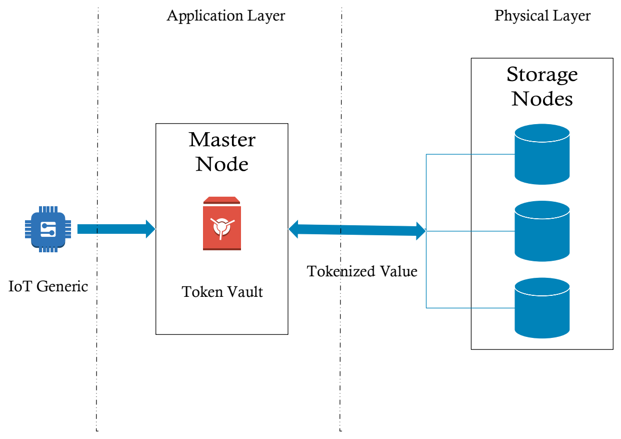


Fig. 4. Tokenization process and Token Mapping

token creation of the same data. Figure 5 depicts the model where there are different TV for each SN. Storage nodes should be mapped along with the tokenization process with a view to ensuring further data token search and retrieval.

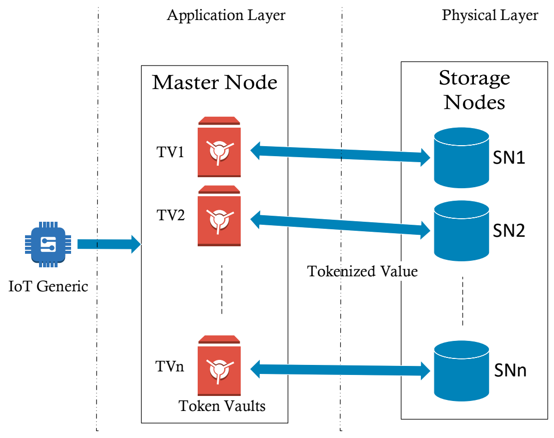


Fig. 5. Different Token vault per storage for optimization

6) Firewall, Rich rules and data policies: Policies and rich rules are managed through multiple virtual firewalls. The system firewall primarily check the incoming and outgoing packets, then the virtual firewalls check packet data before allowing access to tenants and finally the operating system of the instances check validities with it's default firewall.

7) Inbound and Outbound Packet Control through ACL: An access control list (ACL) is a list that contains access control entries (ACE). Each ACE identifies a trusted source and defines the access rights (allow, deny or audit) for that node. Also resource access hierarchy is managed by the ACL. To control the traffic, inbound and outbound ACL is applied.

The ACL is inbound when the node receives packet and it is outbound when the node maps and routes a tokenized packet to the storage nodes through outbound interface.

8) Job Tracker and Task Tracker: Storage node mapper maps the storage nodes and a job tracker will track the mapping and monitor the logs just like the HDFS. A task tracker stays inside the storage nodes and return the requested tokenized data back to the master node according to the requested $device_{ID}$ where job tracker will pass the data to be converted into plain texts with the help of token vault.

9) Web Application Environment: Master node is able to provide secure access to data as well as create temporary web application environment so that applications can process data earlier before leaving the master node. Master node can host applications inside a tenant for ensuring better privacy towards data by computing within a virtual private environment.

10) IoT Certificate: Communication between IoT device and cloud is protected through the use of X.509 certificates. To introduce a device to the cloud, normally cloud provider generates this X.509 certificate for the device. Certificates must be activated prior to use.

B. System processes

1) Device and Data validation of incoming packet: Master node checks the validity of a sender or IoT device using the X.509 certificate and record of registered device lists. Also checks the validity of data packet according to the policies. If ok, the data packet is accepted for segregating the data, tokenizing and mapping to storage afterwards.

2) Data segregation: Both the master node and end sensor device node must have an prior understanding on which data transmission protocol they are going to use for keeping data inside a packet and how the data are segregated inside master node. Data are kept in a temporary buffer after segregation waiting to be mined and tokenized for a particular time stamp.

3) Data tokenization and mining: Data are tokenized from the buffer with the help of token vault according to the policies. Also collected data are mined initially in order to convert the big data into small ones. For mining, we calculated 3M (Mean, Median, Mode) values for a certain range. Then all the derived data are tokenized and put inside a $Token$ that waiting to be mapped and stored in distributed storage nodes. Of course the data are destroyed just after they are tokenized.

4) Data mapping towards storage nodes: Generated $Token$ is now mapped with logging the time stamp, device id and token id accordingly. Then the tokens are sent towards the distributed storage nodes. As the $Token$s are stored directly inside the storage, no encryption or decryption procedure is needed while transmitting the tokens.

5) Further mining of stored data: For further mining, at first the stored $Token$s are sent back to the master node where the tokenized data is converted into plain data for further processing. So, if an application needs to access the data, it will request the master node to provide a secure access to plain data.

Algorithm 1: Algorithm of data mapping towards distributed storage node for a single IoT device

```
System Initialization;
Receive data packets from end nodes;
if $Device_{ID}$ , $IoT_{cert}$ = $True$ then
    Set $Time_{Period}$;
    while data receiving do
        Classify data according to minimal range;
        Create buffer for each range;
        for $Range_{each}$ do
            Collect $Token_{range}$;
            Set $Token_{ID}$;
            while $Time_{Period}$ do
                Count $Data_{freq}$;
                Calculate $Mean$, $Median$, $Mode$;
                Create $TOKEN$;
            end
            Map $TOKEN$;
            Send $TOKEN$ towards storage node;
        end
    end
end
else
    Discard Packet
end
```

IV. Tokenization

A. Token Format

The entire $TOKEN$ takes a $Device_{ID}$, the current $Token_{ID}$, the $Token_{range}$ for each range, the current $Time_{Period}$, tokenized mined data [HEX($Frequency_{data}$), $Token_{MEAN}$, $Token_{MEDIAN}$, $Token_{MODE}$] and some start-ending bits concatenated altogether produces a token, which contains the message in a form that can't be read or altered without the help of the token vault.

So, a complete token is the concatenation of the following fields which is be sent towards storage nodes:

$$\begin{aligned} TOKEN = Bits_{START}||Device_{ID}||Token_{ID} \\ ||Time_{Period}||Token_{range} \\ ||HEX(Frequency_{data})||Token_{MEAN} \\ ||Token_{MEDIAN}||Token_{MODE}||Bits_{END} \end{aligned}$$

- $Bits_{START}$, 16 bits
- $device_{ID}$, 256 bits
- $Token_{ID}$, 256 bits
- $Time_{Period}$, 64 bits
- $Token_{range}$, variable length, e.g. 32 bits
- HEX($Frequency_{data}$), 32 bits
- $Token_{MEAN}$, 8 bits
- $Token_{MEDIAN}$, 8 bits
- $Token_{MODE}$, 8 bits
- $Bits_{END}$, 16 bits

Algorithm 2: Algorithm of data retrieving for further mining from distributed storage nodes

```
System Initialization;
while $Request_{App}$ do
    Define $device_{Id}$;
    for each $device_{Id}$ do
        Backtrack $Map_{Token}$, $Map_{Storage}$;
        Retrieves $TOKEN$ from distributed storage;
        Reconstructs $Data_{Plain}$ from $TOKEN$;
        Send $Data_{Plain}$ towards application;
    end
end
```

The start and ending bits determines the total number of fields as well as mentions the size of fields which are of variable lengths. The sizes mentioned here are provided as an example; it can be changed according to policies. So, the concatenated $TOKEN$ is a string of hexadecimal values.

B. Data Storing and Retrieving Algorithms

According to the work, data storing and mapping process in distributed storage nodes is defined in Algorithm 1 and retrieving process of data for further mining is defined in Algorithm 2

V. Implementation and Results

A. Token Generation

Data Range	Example Token
1-5	A1B1C
6-10	E1F2C
11-15	H1I6D
16-20	J2V5D
21-25	K8F4A
26-30	G6F8Z
31-35	K7D5S
36-40	H9L3R
41-45	K9U0D
46-50	L7C5E

TABLE I
TOKEN FOR DATA RANGE

Floor(Value)%5	Example Token
1	A
2	B
3	C
4	D
0	E

TABLE II
TOKEN FOR REMAINDER VALUE

Suppose a sensor device has sent following demo data (Max:50, Min:1) for a particular $TIME-STAMP$:

$Data$ = [3, 1, 2,7, 48, 20,9, 45, 1, 7, 29, 24, 2, 1, 26, 40, 21, 7, 5, 37, 29, 1, 3, 4, 12, 19, 12, 41, 5, 32, 4, 17, 2, 26, 50, 17, 8, 15, 31, 8, 14, 2, 7, 14, 13, 27, 9, 4, 43, 4, 10, 18, 20, 6, 33, 45, 30, 4, 3, 28, 2,37, 38, 20, 4, 2, 5, 37, 49, 35, 10, 43, 12, 40, 22, 39, 17,48, 2, 3, 1, 45, 2, 19, 1, 24, 4, 40, 16, 36, 15, 16, 37, 25, 19, 42, 23, 30, 48, 33, 46, 19, 28]
So, 10 new tokens will be generated as there are 10 ranges within $1-50$ for that particular $TIME-STAMP$.

B. Token Analysis

Suppose, we will check the first and fifth token of which the data range are (1-5) and (21-25). For the data range (1-5), we get $Mean = 2.75$, $Median = 2.5$ and $Mode = 2$. After rounding the values, we get 3,2 and 2 accordingly. As 5 is the difference between start and ending value of a range, we will divide the results with it. Now, dividing by 5 the remainders remain the same as $(3\%5) = 3$, $(2\%5) = 2$ and $(2\%5) = 2$. Then we get the token values from Table II.
For the data range (21-25), the derived values are $Mean = 23.16$, $Median = 23.5$ and $Mode = 24$. After rounding, the values we get are 23, 23 and 24 accordingly. Now, after dividing by 5, the remainders are 3, 3 and 4 for which the token values are already ready at Table II as usual.

Range	Example Token	Mean	Median	Mode
1-5	A1B1C	$3 \Rightarrow C$	$2 \Rightarrow B$	$2 \Rightarrow B$
21-25	K8F4A	$3 \Rightarrow C$	$4 \Rightarrow D$	$4 \Rightarrow D$

TABLE III
TOKEN FOR DATA RANGE

C. Comparison between Tokenization and AES-128

For a certain range, data frequency can vary from time to time. Figure 6 shows space optimization in case of tokenization and AES-128 EAX mode(Encryption and Authentication). Whatever the data frequency is, token size always remains constant while performance and storage spaces become more costly with increasing data frequency for a certain range in case of encryption e.g. linear increment in case of AES. For data range $1-5$, frequency is 28 and AES-128 generated about 924 (33 Bytes for each digit) Bytes while the tokenized value is still 87 Bytes as still we are counting only the 3M values.

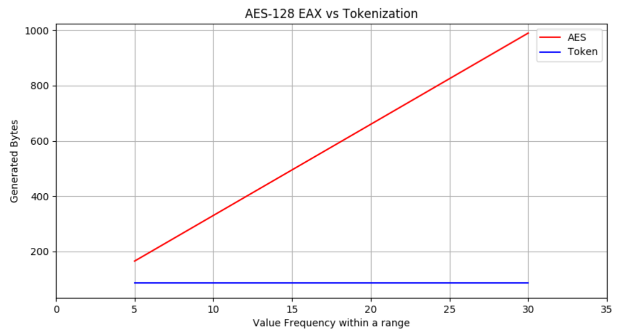


Fig. 6. Space optimization (Token vs AES-128 EAX mode)

VI. Discussion

A. Advantages

1) Optimization in Computation and Storage: Naturally big data needs huge computation power and storage capabilities in optimized way. Ciphering deciphering processes and key generation or management is costly both in computing or storage purposes. Tokenization can easily surpass encryption in optimizing these costs.

2) Additional Requirement of Encryption is unnecessary: As the data are mapped with random tokenized value, no additional encryption method or key management is required hereby so that user won't need to worry about cross layer securities. Also the storage nodes won't need any encryption.

3) No possibility of Insider threat or data breach: The proposed work totally preserves the confidentiality of data as because the tokenized data is impossible to guess without access in token vault so that compromised data won't affect the privacy of data.

4) Availability of Data: Mapping same data towards multiple nodes like Hadoop system ensures the availability of data 24x7. One more benefit is, same data is stored with different values in different nodes that make data more anonymous.

B. Potential Application areas

The work is primarily designed for a smart integration of cloud and IoT in a secure manner. And with the rapid increase in the presence of smart things and future internet technologies a large amount of data is generated which needs to be properly managed and analyzed for various applications using a structured and integrated secure approach. However, such information utilization requires appropriate cloud technologies to collect, store, analyze and visualize large amounts of data from the city environment, citizens and various departments to generate new knowledge and support decision making. Future Health-care, Market Basket Analysis, Fraud Detection, Intrusion Detection, Financial Banking, Research Analysis or Bio-Informatics; everywhere data mining has been an essential need to manage and mine Big Data. In that case, tokenized data can ensure better performance and storage optimization.

C. Scope and Limitations

The primary limitation of the proposed model is only applicable for a smart system where data is homogeneous and needs calculations with approximate values. And increasing data range can affect the performance; although, recent technological advancement on computation and storage optimization can deal with the problem with ease. Also the primary focus should be to provide top level security towards the token vaults and data maps. Once they are compromised together, the data can be detokenized easily from storage nodes.

D. Future Work

Fog computing is a recent framework where prior calculations are done at the end nodes. The proposed work can be merged with Fog computing for better performance. Policies can be developed for the classified data. Also tokenization process need to be developed for various types of data.

VII. Conclusion

The work on security and optimization of data in cloud storage through prior data analysis, tokenization and mapping ensure the data confidentiality to the very end. The 3M values along with the frequency for a particular period can deliver approximate values for further mining. Then tokenization provides maximum optimization of the storage spaces. Data breaches and insider threats are not a concern anymore as compromised or leaked token data cannot be detokenized without accessing the token vault. The architecture has been designed basically for the homogeneous structured data of velocity, veracity and value which introduces the big data world created by IoT or other sensor networks where approximate values are needed for further mining. In that case, the presented work can provide complete data privacy as well as optimize storage spaces through applying the combined approach of tokenization and mining.

References

[1] Dimitrios Zissis and Dimitrios Lekkas, *Addressing cloud computing security issues*, Future Generation computer systems 28.3 (2012): 583-592.

[2] Keiko Hashizume, David G Rosado, Eduardo Fernndez-Medina and Eduardo B Fernandez, *An analysis of security issues for cloud computing*, Journal of Internet Services and Applications 4.1 (2013): 5.

[3] S Subashini and Veeraruna Kavitha, *A survey on security issues in service delivery models of cloud computing*, Journal of network and computer applications 34.1 (2011): 1-11.

[4] "Security Guidance for Critical Areas of Focus in Cloud Computing V3.0" [Online] Available: https://downloads.cloudsecurityalliance.org/assets/research/security-guidance/csaguide.v3.0.pdf (Date last accessed 14-July-2017)

[5] Sophia Yakoubov, Vijay Gadepally, Nabil Schear, Emily Shen, and Arkady Yerukhimovich, *A survey of cryptographic approaches to securing big-data analytics in the cloud*, In High Performance Extreme Computing Conference (HPEC), 2014 IEEE, pp. 1-6. IEEE, 2014.

[6] Joonsang Baek, Quang Hieu Vu, Joseph K. Liu, Xinyi Huang, and Yang Xiang, *A secure cloud computing based framework for big data information management of smart grid*, IEEE transactions on cloud computing 3, no. 2 (2015): 233-244.

[7] Alfredo Cuzzocrea, *Privacy and security of big data: current challenges and future research perspectives*, Proceedings of the First International Workshop on Privacy and Security of Big Data. ACM, 2014.

[8] Jeong-Min Do, You-Jin Song, and Namje Park, *Attribute based proxy re-encryption for data confidentiality in cloud computing environments*, Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on. IEEE, 2011.

[9] Krishna PN Puttaswamy, Christopher Kruegel, and Ben Y. Zhao *Silverline: toward data confidentiality in storage-intensive cloud applications*, Proceedings of the 2nd ACM Symposium on Cloud Computing. ACM, 2011.

[10] Wassim Itani, Ayman Kayssi, and Ali Chehab, *Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures*, Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on. IEEE, 2009.

[11] "Big Data Security and Privacy Handbook : Cloud Security Alliance" [Online] Available: https://cloudsecurityalliance.org/download/big-data-security-and-privacy-handbook (Date last accessed 14-July-2017).

[12] "Big Data Security ENISA" [Online] Available: https://www.enisa.europa.eu/publications/big-data-security (Date last accessed 14-July-2017)

[13] "Encryption at Rest in Google Cloud Platform — Documentation — Google Cloud Platform" [Online] Available: https://cloud.google.com/security/encryption-at-rest/default-encryption/ (Date last accessed 14-July-2017)

[14] "Microsoft Trust Center — Home" [Online] Available: https://www.microsoft.com/en-us/trustcenter (Date last accessed 14-July-2017)