JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 Vol.-15, No.-4, April (2020) pp 294-312 ISSN (Print) 0973-8975

FLEXIBLE SCHEME FOR PROTECTING BIG DATA AND ENABLE SEARCH AND MODIFICATIONS OVER ENCRYPTED DATA DIRECTLY

Sirisha N[1], K. V. D. Kiran[2]

[1,2]Department of Computer Science and Engineering, KoneruLakshmaiah Education Foundation, Vaddeswaram, AP, India.

[1]Department of Computer Science and Engineering, MLR Institute of Technology,Dundigal, Hyderabad, India

[1]grandhishirisha@gmail.com

https://doi.org/10.26782/jmcms.2020.04.00022

Abstract

Secure data storage and retrieval is essential to safeguard data from different kinds of attacks. It is part of information security which enables a system to avoid unauthorized access to data. The data storage destinations are diversified which includes the latest Internet computing phenomenon known as cloud computing as well. Whatever be the storage destination, cryptographic primitives are widely used to protect data from malicious attacks. There are other methods like auditing for data integrity. However, cryptography is the technique which has witnessed many variants of algorithms. However, most of the cryptographic algorithms do not support search and data modifications directly on the encrypted data. Homomorphic encryption and its variants showed promising solution towards flexibility in data dynamics. Motivated by this cryptographic technique, in this paper we proposed an algorithm known as Flexible Data Encryption (FDE) which supports encryption, decryption, search operation directly on encrypted data besides allowing modifications. This improves performance and flexibility in data management activities. Moreover, the proposed algorithm supports different kinds of data like relational and non-relational data. The proposed big data security methodology uses Jalastic cloud as the storage destination. Empirical results revealed that the proposed algorithm outperforms baseline cryptographic algorithms.

Keywords: Big data, big data security, Jelastic cloud, flexible encryption, homomorphic encryption

I. Introduction

Cryptography is the widely used security mechanism. It has been helping to protect information systems from malicious attacks. With respect to data protection when it is at rest, in transit and when it is being used, cryptographic techniques are great choice to safeguard data. Prior to understanding problem with traditional cryptographic methods, they are briefly described here. Cryptography is of many types such as symmetric, asymmetric and quantum cryptography. Quantum cryptography is not described in this paper. Symmetric cryptography, as the name indicates, uses same key both operations like encryption and decryption. However, it causes security issues as the key needs to be sent to the intended recipient. In order to overcome this issue, asymmetric cryptography came into existence. According to this category of cryptography, two keys (private key and public key pair) are pre-distributed to the parties that participate in secure communications. The public key is exposed to all parties while the private key remains secret. The data is encrypted with the receiver's public key and the encrypted data can be decrypted by the private key of receiver. This has eliminated the act of sharing key and thus providing better security then symmetric cryptography. The two kinds of cryptography are illustrated in Figure 1.

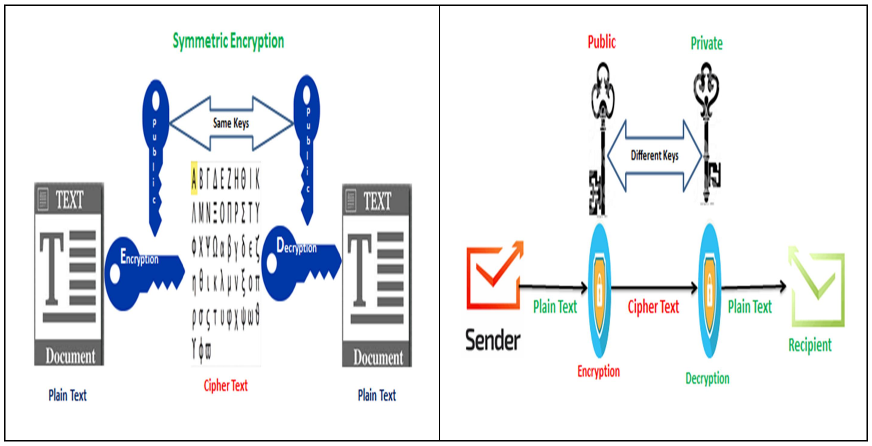


Fig. 1: Symmetric and asymmetric cryptography mechanisms

The traditional cryptographic algorithms of both categories are having limitations when used with public cloud. First, they are not light weight algorithms. Second, they do not entertain modifications and search operations on the cipher text directly. Therefore, it is essential to have an alternative for big data security and outsource big data to cloud with flexible encryption mechanisms. As explored in [XXIX], [VII] and [V], there is an encryption mechanism known as Homomorphic Encryption (HE) [III], [VI](described in Section 4) which is flexible when compared with the traditional security measures. The variant of HE known as evolutionary HE is employed in [XXIX] for privacy and security of big data. Kuzu*et al*. [IX] explored a method based on HE for distributed search over big data that has been encrypted and outsourced to cloud. The functions behind HE and its applications are illustrated in [XV].

The concept of Fully Homomorphic Encryption (FHE) is explored in [X] while [IV] and [XXIX] have implementation of HE for security of big data. HE with GPU environments [XXXI],[XII], FHE for FPGA [XV], HE for databases [VIII], HE for behaviour advertising [XX] are some of the applications where big data is secured with HE. From the literature it is understood that HE[XXI] and FHE are widely used security mechanisms[XVII]. However, when it comes to cloud security with different parties involved, we found that there is need for a comprehensive framework that provide security to different formats of big data. Therefore, in this paper, we proposed such framework. Our contributions are as follows.

1. A framework is proposed for secure outsourcing of big data to cloud computing environment besides performing search and data modifications directly on the cipher text stored in cloud.
2. An algorithm named Flexible and Efficient Encryption (FEE) is proposed as part of the proposed framework for flexible encryption.
3. A prototype is developed using Java platformto show how the proposed FEE algorithm works.

The remainder of the paper is structured as follows. Section 2 reviews literature on HE based solutions for big data[XXII]. Section 3 presents big data security issues. Section 4 presents the details of HE. Section 5 provides the proposed framework with underlying algorithm[XXIII]. Section 6 presents experimental setup. Section 7 presents results of empirical study. Section 8concludes the paper and gives directions for future work[XXIV].

II. Related Works

Review of literature on big data security with flexible encryption schemes is provided here. Jiang *et al*. [VII] proposed a methodology for security of big data and making queries on the outsourced data with privacy preserved. Search is carried out based on Locality Sensitive Hashing (LSH) which is multi-dimensional in nature. hen*et al*. [V] on the other hand provided scheme with a hierarchical approach in searching big data cipher text. They supported a ranked search with multiple keywords. Li *et al*. [XV] proposed the notion of intelligent cryptography that is known as Security Aware Efficient Distributed Storage (SE-EDS). It could reduce time complexity in security operations. Elhosen*et al*. [II] proposed machine learning algorithms that act on encrypted data. It could provide secure data analytics based on HE. They intended to improve the scheme with distribution of workload between cloud and thick client.

The need for security and privacy of big data is emphasized in [I] the realization of the same with flexible data retrieval and processing with HE is studied in [XXV]. Fun and Samsudin[XXV] reviewed techniques based on HE for computations on the outsourced big data. Rauthan and Vaisla [VIII] proposed a service associated with cloud known as Database as a Service (DBaaS) that focused on security and data with flexible encryption mechanisms. They proposed a database known as Verifiable Reliable Secure-Database (VRS-DB) for implementation of privacy and security. Hariharan*et al*. [XIX] proposed a security approach to big data based on data perturbation with nested clustering mechanism. They used Spark as the framework for big data security evaluation. Tourky*et al*. [XXVII] opined that HE is the cryptography's holy grail and it supported sophisticated algorithms based on it from time to time. It revealed the significance of HE for flexible data encryption and operations.

Peter Pietzuch*et al*. [XVIII] focused on stream processing pertaining to big data with trusted security mechanisms. They created environment known as Trusted Execution Environment (TEE) for safeguarding big data while it is subjected to data analytics. Kim *et al*. [XI] focused on big data of FPGA clouds for data protection. They created a product known as SafeDB that provides systematic security to data stored. Zhiqiang and Longjun [XXXIII] focused on the privacy and security of big data processing. They investigated on privacy of big data when it is being processed. They also employed differential privacy. Rawat*et al*. [XX] and Yadav*et al*. [XXXII] focused on cyber security of big data and security and privacy issues on big data respectively. The review of literature here provided insights to the effect that there is need for a more flexible framework for handing different kinds of big data, secure it and provides operations on the cipher text. Such framework is realized in this paper.

III. Big Data Security Issues

With the emergence of big data[XXVI], enterprises are using every piece of data to be stored and analysed or mined for comprehensive business intelligence. Thus large amounts of data are being saved to public cloud. There are different sources of big data such as weather applications, social networks and sensor networks to mention few. As data is in petabytes scale, usage of cloud resources is recommended. Organizations in different domains, countries and governments are preferring cloud resources for storing big data[XXVII]. Outsourcing data and computations to public could may lead to security issues. Thus, unless security is ensured, the data owners suspect cloud resources due to many reasons for losing data or integrity of data. There are many traditional security[XXVIII] issues such as insider and outsider attacks. Besides, cloud storage with big data in place has led to many security issues[XXIX]. As per Cloud Security Alliance (CSA), there are many big data issues. They are categorized into data management issues, data security issues, infrastructure security[XXX] issues and privacy issues.

As presented in Figure 2, there are four categories of security problems pertaining to cloud and big data. Infrastructure security refers to the security[XI] related to infrastructure which may have hidden security[XIII] problems or vulnerabilities to various kinds of attacks. Data management challenges include data storage, retrieval and sharing of the same with end to end security[XIV]. Data privacy refers to the privacy (protecting sensitive data) of big data. The data integrity is also given importance as the consistency, correctness and completeness are essential for outsourced data. Reactive security refers to the security measures that take necessary steps against intrusion

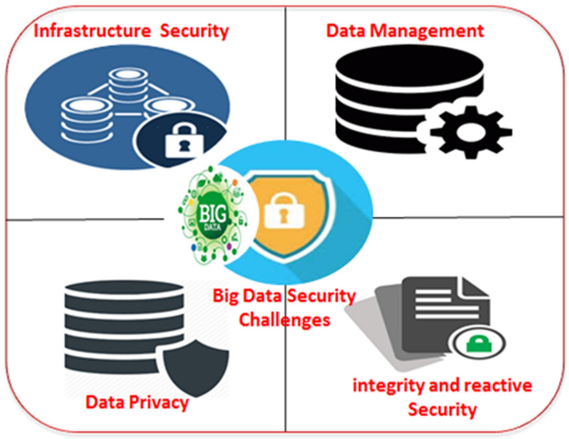


Fig. 2: Big data security challenges as per CSA

IV. Homomorphic Encryption

Homomorphic Encryption (HE)is not similar to that of conventional cryptography. HEhas provision for performing operations directly on the cipher text. The traditional algorithms like RSA, DES and AES do not support this kind of feature. Moreover, HE is flexible and suitable for cloud storage and retrieval without compromising security. When data needs to be searched, there is no need for decryption and thus, the operations on the cipher text do not cause much overhead. As explored in [XXVI], there are different kinds of HE like multiplicative HE and additive HE. The approach used by them are different hence the name.

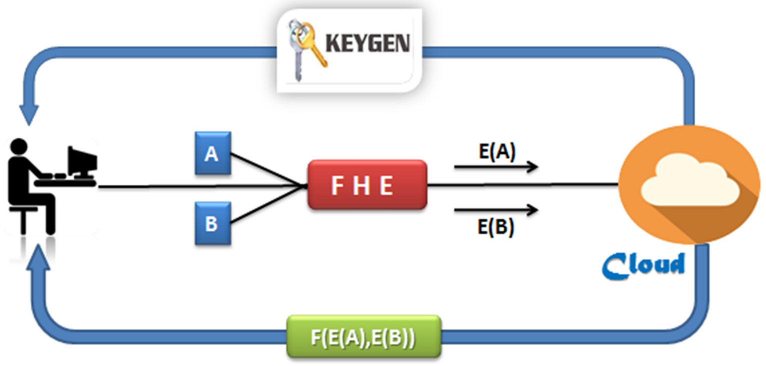


Fig. 3: Modus operand of HE

The overview of HE is presented in Figure 3. As studied in [XVI] and [XXVIII] HE has different operations that are essential. The key generation process is used to have dynamically generated keys. The storage mechanism is associated with cloud. Request is made by end users and data owners in general. Evaluation is the procedure followed in server. Server gives response against queries. The encryption and decryption procedures do the conversion of plain text to cipher text and vice versa respectively.

Table 1: Common phases of HE

Operation	**Description**
Key generation	Two keys known as public key (**pk**) and secret key (**sk**) are generated in this phase.
Encryption	The client program needs to encrypt data with a key and then encrypted data and pk are sent to server.
Storage	Storage in cloud includes pk and encrypted data.
Request	Made by client program to perform different operations on the cloud server.
Evaluation	Server evaluates requests prior to giving response.
Response	Server gives response to client program.
Decryption	Client uses sk to decrypt the encrypted results coming from server.

As presented in Table 1, HE involves in many operations. The operations are part of different kinds of HE. Partially Homomorphic Encryption (PHE), Fully Homomorphic Encryption (FHE) and SomewhatHomomorphic Encryption (SWHE) are the different variants of the HE.

V. Proposed Framework for Flexible Data Encryption

Unlike traditional encryption techniques, the proposed big data protection framework throws light into flexible encryption mechanism that supports data modifications and dynamic search on the encrypted data directly. More details on the framework, underlying algorithm and data dynamics are provided in the following sub sections.

V.i. Problem Definition

Provided different kinds of data (big data), design an encryption algorithm that promotes search and modifications directly on the data that has been encrypted. It is the problem statement precisely. The conventional methods used for cryptography do not support data modifications directly on the encrypted data. This is an important drawback in cloud computing environments where users need flexibility to store, retrieve, search and manipulate data. These operations are not directly supported on the traditional schemes. When such operations are allowed, it will make the framework of data management more flexible besides making it intuitive to users. This is the motivation behind taking up this work.

V.ii. The Framework

The proposed framework is based on FHE. The framework deals with big data security in terms of flexible encryption and decryption schemes besides search and changes made to data on the cipher text directly. Since big data is stored and managed in public cloud, the proposed framework includes cloud environment as well. Cloud Service Provider (CSP) is one of the parties involved in the security scheme. Data owners can outsource large volumes of data and see that the data is secure and supports flexibility with direct operations on the cipher text.

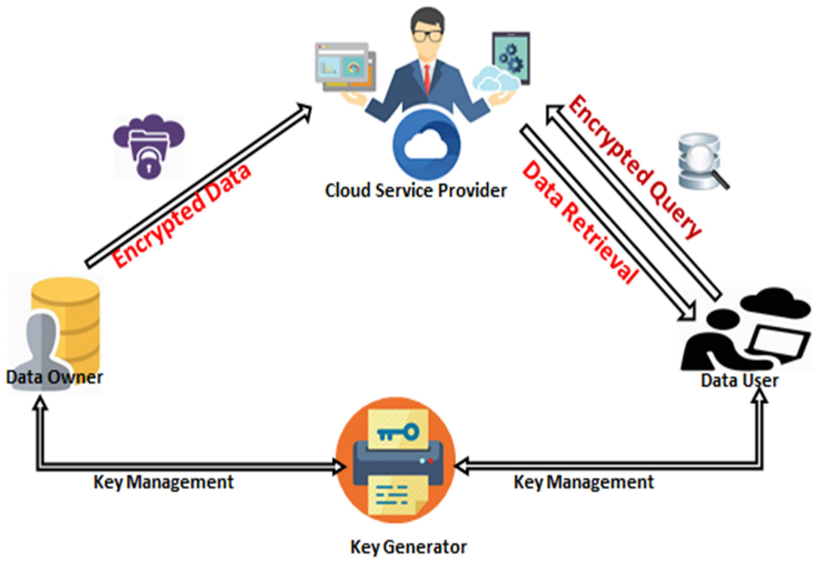


Fig. 4: Architectural overview of the proposed framework

As shown in Figure 4, the framework has different parties like CSP, data owner, data user and key generator. It employed the proposed algorithm based on FHE for data storage and dynamics with complete security. Arbitrary changes to cipher text are made directly with required flexibility on the outsourced big data. Data owner owns data and performs encryption prior to outsourcing data. Data owner may be an organization or a person. Data user on the other hand is subscribed to data present in the cloud. With access permissions, this user can get data as needed. The queries made by the user are encrypted for security reasons. The results also come in cipher text prior to decryption. Both the user and data owner use the prototype application developed using Java programming language with intuitive interface. They need not be experts in cryptographic primitives.

CSP is an important entity in the given framework where computing resources are utilized on demand. This is Jelastic cloud IaaS infrastructure used in the implementation of the framework in this paper. It is crucial for storage and retrieval of data with built-in provisions. Key generator is another entity involved in the system. it is responsible to provide keys needed by the users. The system is based on FHE that satisfies the objective of this paper in providing a flexible and efficient encryption mechanism. In this framework, CSP cannot get data and cannot interpret data and query results. Nothing is allowed by the server except acting as a repository for storage.

V.iii. Proposed Algorithm

The algorithm proposed is known as Flexible and Efficient Encryption algorithm. The algorithm has three crucial phases known as key generation, performing flexible encryption and decryption. The keys are generated and distributed to intended users and they are used for performing outsourcing big data, searching, and data modifications.

```
Algorithm: Flexible and Efficient Encryption (FEE)
Choose two prime numbers p and q
Generate key, KeyGen(p,q), where the inputs p,q $\epsilon P$
Multiply p and q to arrive at N, N=p*q and then choose g $\epsilon Z_{n^2}$ such that
gcd(L($g^{\lambda}$ $mod\ n^2$), n) =1 with L(u)=$\frac{u-1}{n}$
pk =(n,g) and sk=(p,q) are public and secret keys
Encryption: C=$g^m r^n$mod $n^2$
Provided two ciphers $c_1 and\ c_2$
Choose $m_1$, $r_1$as an arbitrary whole number for HE
        $c_1 = g^{m_1} {r_1}^n$mod $n^2$
Choose $m_2$, $r_2$as an arbitrary whole number for HE
$c_2 = g^{m_2} {r_2}^n$mod $n^2$
Then   $c_1 c_2$= $g^{m_1+m_2}(r_1 r_2)^n$ mod $n^2$
Decryption:  m=$\frac{L(c^{\lambda} mod\ n^2)}{L(g^{\lambda} mod\ n^2)}$ mod n
```

Algorithm 1:Flexible and efficient encryption algorithm

As presented in Algorithm 1, there are procedures for encryption and decryption based on FHE mechanism. Once data is converted into cipher text, it is outsourced to cloud as per the framework. Such data can be searched directly without converting cipher text into plain text. Besides, it also helps in making changes directly. Section 3.4 provides more details on the data dynamics process. The big data converted into cipher text can withstand various kinds of attacks. Besides, such cipher text allows search and changes directly. As traditional algorithms like RSA do not support this flexibility, the proposed algorithm is defined based on FHE that supports it. Even user queries and the results returned by server are converted to cipher text for higher level of security. It not only adds flexibility and intuitiveness from user point of view, but also it reduces burden on the server. It leads to deterioration of overhead.

The empirical results with the proposed encryption algorithm are observed using the prototype application built.

V.iv. Predicate Evaluation in Data Dynamics

Data once outsourced, that can be modified using different queries that make use of predicates. This section provides evaluation of such predicates to achieve data dynamics. FHE based scheme makes use of properties like x and y (finite integer domain) and their binary counterparts like b_x and b_y. Provided this, there are many functions like multiplication, subtraction and addition. When function space is denoted as f^*, there are different properties that can be used to have different instances of f^*. Property P1 refers to addition of integers. It is denoted as Enc(x) + Enc(y) = Enc(x + y). Similarly property P2 refers to multiplication of integers which is denoted as Enc(x) ∗Enc(y) = Enc(x ∗ y). Property P3 (A) refers to subtraction of integers. It is denoted as Enc(x) − Enc(y) = Enc(x − y). The property named P3(B) refers to subtraction on binary numbers. It is mathematically represented as Enc(bx) − Enc(by) = Enc(bx − by). The property named P4 refers to a bitwise AND operation. It is denoted as Enc(bx) ∧Enc(by) = Enc(bx∧ by). The property named P5 refers to bitwise OR operation. It is denoted as Enc(bx) ∨Enc(by) = Enc(bx∨ by). Another property named P6 refers to bitwise NOT operation which is denoted as Enc(1) $\oplus[Enc(bx)]_i = Enc(\overline{[bx]}_i)$. Under HE, P1-P6 are used to achieve flexible encryption. As cloud provides very large buffer arbitrary values are kept in buffer. When a set of values are given such that $val_j(1 \leq j \leq N)$ and all values are obtained insuch a way that (val_j= x). In order to achieve this, the values are initialized in buffer.

Evaluation of ((Enc(val_j)−Enc(x) == Enc(0)) ? Enc(1) : Enc(0)) on server needs to use encrypted repsenatns. In case of binary representations in encrypted format Enc($bval_j$) and Enc(bx) for Enc(val_j) and Enc(x), the similar evaluations can be reused. Such evaluations can be represented as binary circuits. For the given condition ((Enc(val_j) − Enc(x) == Enc(0)) ? Enc(1) : Enc(0)), the equivalent circuit representation is as in Eq. 1.

$$\wedge_{i=2}^{n+1}(\mathrm{Enc}(1) \oplus ([Enc(bval_j)]_i - [Enc(bx)]_i)) \tag{1}$$

In the same fashion, when a range query needs to be evaluated such as ((Enc(val_j) − Enc(x) < Enc(0)) ? Enc(1) : Enc(0)), the Eq. 2 shows circuit representation.

$$[Enc(bval_j)]_1 - [Enc(bx)]_1 \tag{2}$$

For a range query condition denoted as (val_j)> x), Eq. 3 shows the equivalent circuit operation.

$$\mathrm{Enc}(1) \oplus([Enc(bval_j)]_1 - [Enc(bx)]_1) \tag{3}$$

For every encrypted value Enc(val_j), the predicate evaluation is shown in Eq. 4.

$$\mathrm{Enc}(val_j) * \mathrm{pack}(\text{p-circuit}(\mathrm{Enc}(bval_j))) \tag{4}$$

The above illustrations provide basic understanding of the operations on the encrypted data. The big data is not only secured but also subjected to search and flexible modifications. Section 6 and Section 7 provide experimental setup and results respectively.

VI. Experimental Setup

Experimental setup is made with Jelastic cloud, Java development environment and two kinds of databases such as MY SQL (for storing structure data) and MongoDB (for storing unstructured and semi-structured data). Jalastic cloud is configured to have remote connectivity with public IP address. The cloud environment is limited to storage and retrieval of data. A prototype application is developed with intuitive interface using Java programming language. The application facilitates users to outsource data and also perform search and data modification operations. Figure 5 shows the outline of the environment used for empirical study.

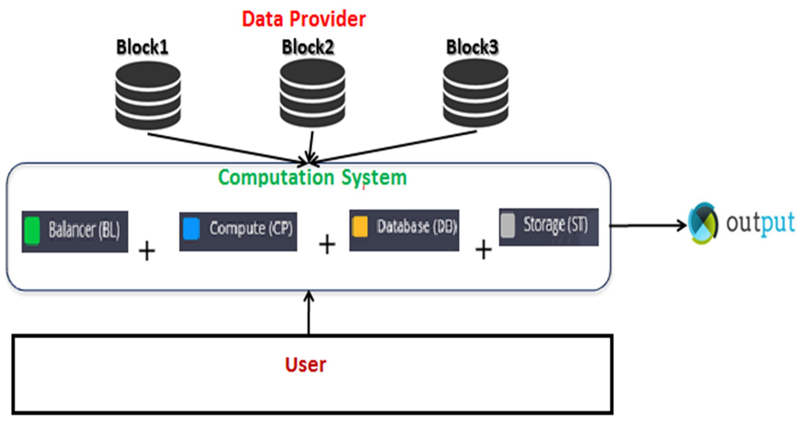
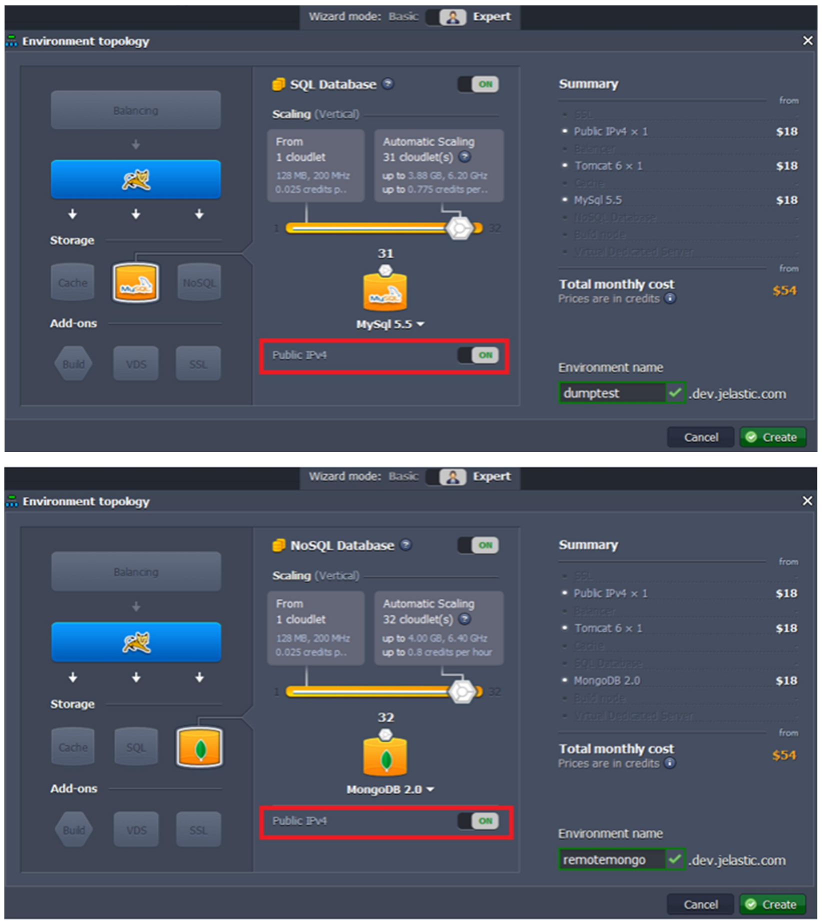


Fig. 5: User interaction with Jelastic cloud

Users or data owners interact with the prototype application built for empirical study. However, the data is outsourced to one of the data centresprovided by Jelastic cloud. The computation system associated with the cloud is made up of balancer, compute, database and storage. The storage part of the environment is made up of MY SQL and MongoDB. The environment is created with two topologies to suit MY SQL and MongoDB.



SQL Database ON
Scaling (Vertical)
From 1 cloudlet — 128 MB, 200 MHz — 0.025 credits p...
Automatic Scaling 31 cloudlet(s) — up to 3.88 GB, 6.20 GHz — up to 0.775 credits per...
31
MySql 5.5
Public IPv4 ON

Summary
Public IPv4 × 1 $18
Tomcat 6 × 1 $18
MySql 5.5 $18
Total monthly cost $54
Prices are in credits
Environment name: dumptest .dev.jelastic.com

NoSQL Database ON
Scaling (Vertical)
From 1 cloudlet — 128 MB, 200 MHz — 0.025 credits p...
Automatic Scaling 32 cloudlet(s) — up to 4.00 GB, 6.40 GHz — up to 0.8 credits per hour
32
MongoDB 2.0
Public IPv4 ON

Summary
Public IPv4 × 1 $18
Tomcat 6 × 1 $18
MongoDB 2.0 $18
Total monthly cost $54
Prices are in credits
Environment name: remotemongo .dev.jelastic.com

Fig.6: Storage configurations for MY SQL (above) and MongoDB (below)

As presented in Figure 6, the storage configurations are made by creating two environment topologies. Each topology provides required environment for a database. The topology shown in Figure 3 (above) is pertaining MY SQL while the topology in Figure 3 (below) is related to MongoDB which is a NoSQL database. These topologies led to the access given to users or data owners to reach the two databases remotely. User application uses JDBC type 4 driven in order to establish remote connectivity through given IP address. When public IP address is used, the prototype application can locate the database servers and the interaction is made possible. The experimental setup helps in execution of the application to show the utility of the proposed security scheme on top of fully homomorphic encryption. It provides security and also flexibility in terms of dynamically searching on encrypted data and making modifications on the encrypted data as well.

VII. Experimental Results

Experiments are carried out with a prototype built using Java programming language. The application is intuitive in nature with Graphical User Interface (GUI) for outsourcing data to public cloud, data encryption, search and data modifications. The application supports connectivity to cloud based MY SQL and MongoDB. The former stores structured data while the latter manages unstructured and semi-structured data formats. The proposed system allows search and data modification operations directly on the encrypted data. The proposed algorithm known as Flexible and Efficient Encryption (FEE) is used to realize the intended functionalities of the system. FEE has security mechanisms based on FHE to achieve flexible and efficient changes made to data in encrypted format. The security scheme is compared with the baseline encryption scheme known as Advanced Encryption Standard (AES). Empirical study is made with the frontend application connecting to two cloud based backbends such as MY SQL and MongoDB. Through public IP address, the application connects to remote data repositories. Empirical study with different sizes of data showed the performance and flexibility of the proposed algorithm.

VII.i. Execution Time Comparison

As the execution time is one of the frequently used performance measures, it is used to evaluate the proposed FEE with a baseline cryptographic solution known as AES. AES is considered to be the baseline algorithm whose results are compared with the FEE. Observations related to execution time are against different size of data.

Table 2: Time taken for cryptographic operations

Data Size (MB)	Execution Time (seconds)			
	Encryption (AES)	Encryption (FEE)	Decryption (AES)	Decryption (FEE)
10	0.9168	0.89909	0.8956	0.7932
50	2.5237	2.3956	1.9879	1.9936
100	2.8948	2.6979	2.6583	2.1267
500	13.7648	12.9997	9.7845	9.1243

As presented in table 2, the encryption time and decryption time are recorded for AES as baseline approach and the proposed FEE algorithm. The results are observed with different data sizes.

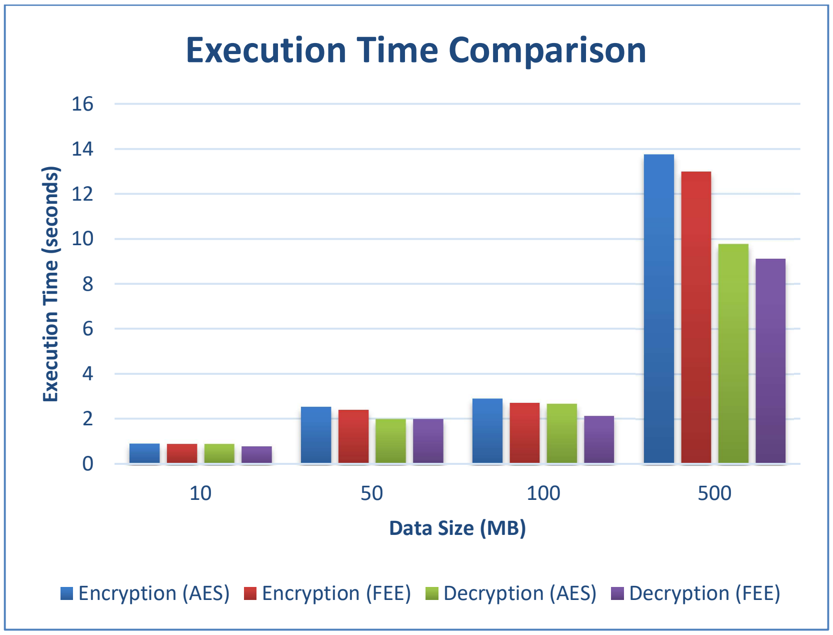


Fig.7: Execution time comparison

As shown in Figure 7, the size of data used for empirical study is provided in horizontal axis..The time taken for encryption of the data and decryption of the data is provided in vertical axis. The baseline algorithm AES and the proposed FEE algorithm are compared in terms of execution time. From the results, it is understood that the size of data has its influence on the time taken for encryption and decryption. As data size is increased, the time taken is increased. The baseline encryption and decryption took more time when compared with FEE. This observation is consistent across all the sizes of data used for empirical study. When data size is 10 MB, the encryption time exhibited by AES is 0.9168 seconds while FEE showed 0.8990 seconds. In the same fashion, the decryption time of AES is 0.8956 while the decryption time of FEE is 0.7932 seconds. The time taken is less than one second. When the data size is 500 MB, AES exhibits 13.7648 and 9.7845 seconds respectively for encryption and decryption. FEE shows 12.9997 and 9.1243 seconds respectively for encryption and decryption. This has revealed the significant improvement of the proposed algorithm in terms of execution time.

VII.ii. Time Taken for Data Upload to Jelastic Cloud

The time taken for uploading data to Jelastic cloud is observed for baseline security algorithm AES and the proposed FEE algorithm. The observations are made with different data sizes. The total time taken for upload includes the time taken for outsourcing data to public cloud including encryption process. The data size is measured in MB while the time is measured in seconds.

Table 3: Data upload time against different sizes of data

Data Size (MB)	Time Taken for Data Upload (seconds)	
	Existing Scheme	**Proposed Scheme**
10	0.6973	0.5678
50	2.8273	2.271
100	4.5873	3.4369
500	17.5373	14.1967

As presented in table 3, the execution time for data upload is recorded for AES as baseline approach and the proposed FEE algorithm. The results are observed with different data sizes.

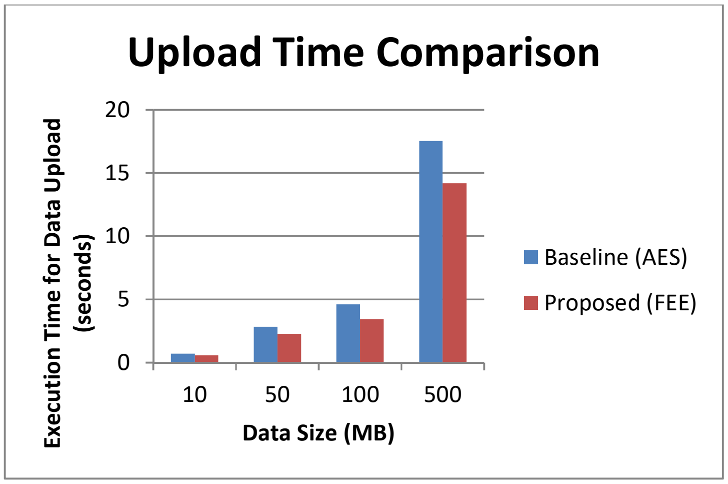


Fig.8: Execution time for data upload against different size of data

As shown in Figure 8, the size of data used for empirical study is provided in horizontal axis. The time taken for data upload is provided in vertical axis. The baseline algorithm AES and the proposed FEE algorithm are compared in terms of execution time for data upload. From the results, it is understood that the size of data has its influence on the time taken for data upload. As data size is increased, the time taken is increased. The baseline algorithm took more time when compared with FEE. This observation is consistent across all the sizes of data used for empirical study. When data size is 10 MB, the upload time taken by AES is 0.6973 seconds while the FEE has taken 0.5678 seconds. When data size is 500 MB, the upload time taken by AES is 17.5373 seconds while the FEE has taken 14.1967 seconds.

VII.iii. Time Taken for Download of Data from Jelastic Cloud

The time taken for downloading data from Jelastic cloud is observed for baseline security algorithm AES and the proposed FEE algorithm. The observations are made with different data sizes. The total time taken for download includes the time taken for downloading data from public cloud and decryption process. The data size is measured in MB while the time is measured in seconds.

Table 4: Execution time for data download (seconds)

Data Size (MB)	**Execution Time for Data Download (sec)**	
	Baseline (AES)	**Proposed (FEE)**
10	0.9169	0.8229
50	2.4348	2.0967
100	3.8048	3.5374
500	13.7648	12.4679

As presented in table 4, the execution time for data download is recorded for AES as baseline approach and the proposed FEE algorithm. The results are observed with different data sizes.

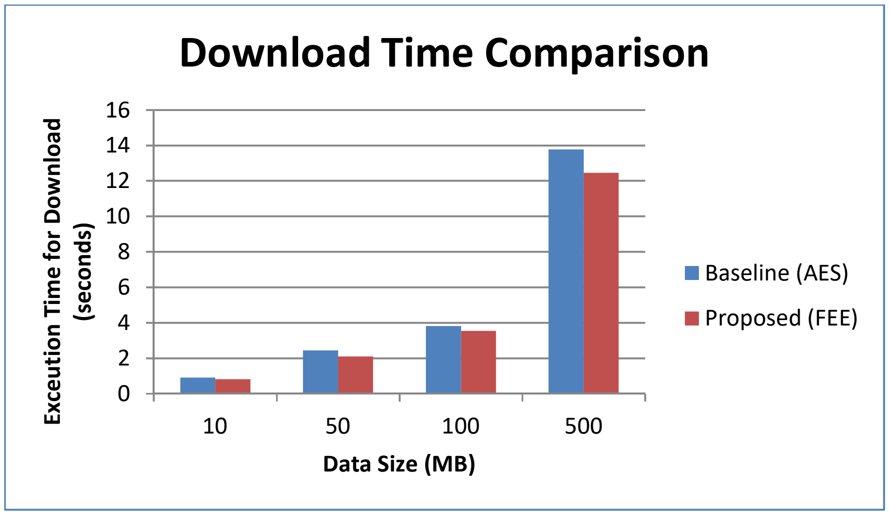


Fig. 9: Total download time against different volumes of data

As shown in Figure 9, the size of data used for empirical study is provided in horizontal axis. The time taken for data download is provided in vertical axis. The baseline algorithm AES and the proposed FEE algorithm are compared in terms of execution time for data upload. From the results, it is understood that the size of data has its influence on the time taken for data download. As data size is increased, the time taken is increased. The baseline algorithm took more time when compared with FEE. This observation is consistent across all the sizes of data used for empirical study. When data size is 10 MB, the download time taken by AES is 0.9169 seconds while the FEE has taken 0.8229 seconds. When data size is 500 MB, the download time taken by AES is 13.7648 seconds while the FEE has taken 14.4679 seconds.

The FEE scheme provides unique results every time. Even when same input is given, the output is rendered differently. In other words, it takes care of uniqueness in choosing a random number and resultant cipher text. When input is given as 123, the random number generated is +qvTW2H6hJJKalaZGJ19Hg== while the cipher text is as follows.

```
647340590292209194227925871318444863715118495520340181789944860
808456312883293664625719823912839885831877026227145618239009775
000860611082936663673286519334086480196545755277748975269611862
105953774255798840050276170528697578883428644860848719810316829
06354684551213299973617854200188304901583492413718273120
```

In another experiment, input is given as same number 123 but its random number r is different. It is +skBW2K6hIFSalaJBJ19Op==. And the cipher text generated is different from that of the first experiment with 123. The resultant cipher text is as follows.

```
161835147573052298556981467829611215928779623880085045447486215
202114078220823416156429955978209971457969256556786404559752443
750215152770734165918321629833521620049136438819437243817402965
526488443563949710012569042632174394720857161215212179952579207
26588671137803324993404463550047076225395873103429568280
```

This has provided evidence of the uniqueness. However, when the two cipher texts are subjected to decryption, interestingly both showed same output that is 123. The rationale behind this is that the original input value is same in both the experiments made.

VIII. Conclusion and Future Work

Securing big data that is in relational or non-relational format is the main focus of this paper. In addition to secure storage and retrieval, it throws light into flexibility in search and data modifications to be carried out on the encrypted data directly. When data is encrypted with traditional cryptographic algorithms like AES (baseline algorithms), it is not possible to perform search operation data modifications on encrypted data storage in any destination including Infrastructure as a Service (IaaS) cloud. Therefore, the baseline algorithms are not flexible while performing data dynamics from a remote client application which provides intuitive interface to end users. To overcome this problem, in this paper, we proposed an algorithm based on homomorphic encryption standard. The algorithm is known as Flexible and Efficient Encryption (FEE) which improves state of the art as it makes the data storage and retrieval flexible and supports relational and non-relational data. In Jelastic cloud, MY SQL is used for relational data storage while non-relational data is maintained in the MongoDB configured. A prototype is built to evaluate the proposed encryption scheme. The empirical results revealed that the proposed system outperforms baseline cryptographic primitives. In future, we intend to investigate on privacy of big data along with security.

References

I. DING, Wenxiu; YAN, Zheng; and DENG, Robert H.. Encrypted data processing with Homomorphic Re-Encryption. (2017). Information Sciences, 35-55.

II. Elhoseny, M., Elminir, H., Riad, A., & Yuan, X. (2016). A secure data routing schema for WSN using Elliptic Curve Cryptography and homomorphic encryption. Journal of King Saud University - Computer and Information Sciences, 28(3), 262–275.

III. Gai, K., &Qiu, M. (2018). Blend Arithmetic Operations on Tensor-Based Fully Homomorphic Encryption Over Real Numbers. IEEE Transactions on Industrial Informatics, 14(8), 3590–3598.

IV. Graepel, T., Lauter, K., &Naehrig, M. (2013). ML Confidential: Machine Learning on Encrypted Data. Information Security and Cryptology – ICISC 2012, 1–21.

V. Hen, C., Zhu, X., Shen, P., & Hu, J. (2014). A hierarchical clustering method for big data oriented ciphertext search. 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). P1-6.

VI. Helsloot, L. J., Tillem, G., &Erkin, Z. (2017). AHEad: Privacy-preserving online behavioural advertising using homomorphic encryption. 2017 IEEE Workshop on Information Forensics and Security (WIFS). P1-6.

VII. Jiang, R., Lu, R., &Choo, K.-K. R. (2018). Achieving high performance and privacy-preserving query over encrypted multidimensional big metering data. Future Generation Computer Systems, 78, 392–401.

VIII. J.S. Rauthan, K.S. Vaisla, VRS-DB: Preserve confidentiality of users' data using encryption approach, Digital Communications and Networks, p1-14.

IX. Kuzu, M., Islam, M. S., &Kantarcioglu, M. (2015). Distributed Search over Encrypted Big Data. Proceedings of the 5th ACM Conference on Data and Application Security and Privacy - CODASPY '15. P1-8.

X. Khedr, Alhassan, et al. "SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers." IEEE Transactions on Computers 65, 9, 2848–2858.

XI. Kim, H.-Y., Myung, R., Hong, B., Yu, H., Suh, T., Xu, L., & Shi, W. (2019). SafeDB: Spark Acceleration on FPGA Clouds with Enclaved Data Processing and Bitstream Protection. 2019 IEEE 12th International Conference on Cloud Computing (CLOUD). P1-8.

XII. KashiSai Prasad, S Pasupathy, "Real-time Data Streaming using Apache Spark on Fully Configured Hadoop Cluster", J.Mech.Cont.& Math. Sci., Vol.-13, No.-5, November-December (2018) Pages 164-176.

XIII. K. Sai Prasad, Dr. S Pasupathy, "Deep Learning Concepts and Libraries Used in Image Analysis and Classification", TEST Engineering & Management, Volume 82, ISSN: 0193 - 4120 Page No. 7907 - 7913.

XIV. K. Sai Prasad &RajenderMiryala "Histopathological Image Classification Using Deep Learning Techniques" International Journal on Emerging Technologies 10(2): 467-473(2019)

XV. Li, Y., Gai, K., Qiu, L., Qiu, M., & Zhao, H. (2017). Intelligent cryptography approach for secure distributed big data storage in cloud computing. Information Sciences, 387, 103–115.

XVI. Maha TEBAA, Said EL HAJII, "Cloud Computing through Homomorphic Encryption", International Journal of Advancements (IJACT), Vol. 8, No. 3, March – April 2017.

XVII. Ogburn, M., Turner, C., &Dahal, P. (2013). Homomorphic Encryption. Procedia Computer Science, 20, 502–509.

XVIII. PeterPietzuch and Valerio Schiavoni. (2019). Using Trusted Execution Environments for Secure Stream Processing of Medical Data, p1-16.

XIX. R.Hariharan, S. Saran Raj and R. Vimala. (2018). A Novel Approach for Privacy Preservation in Bigdata Using Data Perturbation in Nested Clustering in Apache Spark. Journal of Computational and Theoretical Nanoscience. 15 (.), p1-6.

XX. Rawat, D. B., Doku, R., &Garuba, M. (2019). Cybersecurity in Big Data Era: From Securing Big Data to Data-Driven Security. IEEE Transactions on Services Computing, 1–18.

XXI. Shiyuan Wang, DivyakantAgrawal and Amr El Abbadi. (2012). Is Homomorphic Encryption the Holy Grail for Database Queries on Encrypted Data, p1-18.

XXII. Sirisha, N., &Kiran, K. V. D. (2018), "Authorization of Data In Hadoop Using Apache Sentry", International Journal of Engineering and Technology, 7(3), 234-236.

XXIII. Sirisha, N., Kiran, K. V. D., &Karthik, R, (2018), "Hadoop security challenges and its solution using KNOX", Indonesian Journal of Electrical Engineering and Computer Science,12(1), 107-116.

XXIV. Sirisha, N., &Kiran, K. V. D. (2017), "Protection of encroachment on bigdata aspects", International Journal of Mechanical Engineering and Technology, 8(7), 550- 558.

XXV. Tan Soo Fun and AzmanSamsudin. (2016). A Survey of Homomorphic Encryption for Outsourced Big Data Computation. KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS. 10 (8), p1-26.

XXVI. T.Ramaporkalai, "Security Algorithms in Cloud Computing", International Journal of Computer Science Trends and Technology Vol. 5. Issue 2, Mar – Apr 2017.

XXVII. Tourky, D., ElKawkagy, M., &Keshk, A. (2016). Homomorphic encryption the "Holy Grail" of cryptography. 2016 2nd IEEE International Conference on Computer and Communications (ICCC). P1-6.

XXVIII. V. Biksham, D. Vasumathi"Homomorphic Encryption Techniques for securing Data in Cloud Computing: A Survey", International Journal of Computer Applications Vol.160, February 2017.

XXIX. Wei Dai, Doroz, Y., &Sunar, B. (2014). Accelerating NTRU based homomorphic encryption using GPUs. 2014 IEEE High Performance Extreme Computing Conference (HPEC). P1-6.

XXX. Wei Wang,&Xinming Huang. (2013). FPGA implementation of a large-number multiplier for fully homomorphic encryption. 2013 IEEE International Symposium on Circuits and Systems (ISCAS2013). P1-4.

XXXI. Yu, S. (2016). Big Privacy: Challenges and Opportunities of Privacy Study in the Age of Big Data. IEEE Access, 4, 2751–2763.

XXXII. Yadav, D., Maheshwari, D. H., & Chandra, D. U. (2019). Big Data Hadoop: Security and Privacy. SSRN Electronic Journal. P1-8.

XXXIII. Zhiqiang, G., &Longjun, Z. (2017). Privacy Preserving Data Mining on Big Data Computing Platform: Trends and Future. Lecture Notes on Data Engineering and Communications Technologies, 491–502.