Information Sciences 000 (2016) 1–13

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Intelligent cryptography approach for secure distributed big data storage in cloud computing

Yibin Li[a], Keke Gai[b,*], Longfei Qiu[c], Meikang Qiu[b,1], Hui Zhao[d]

[a] School of Computer Science and Technology, Shandong University, China
[b] Department of Computer Science, Pace University, New York City, NY 10038, USA
[c] Nanjing Foreign Language School, Jiangsu, China
[d] Software School, Henan University, Kaifeng, Henan, 475000, China

ARTICLE INFO

Article history:
Received 27 December 2015
Revised 2 September 2016
Accepted 3 September 2016
Available online xxx

Keywords:
Intelligent cryptography
Cybersecurity
Mass distributed storage
Cloud computing
Big data

ABSTRACT

Implementing cloud computing empowers numerous paths for Web-based service offerings to meet diverse needs. However, the data security and privacy has become a critical issue that restricts many cloud applications. One of the major concerns in security and privacy is caused by the fact that cloud operators have chances to reach the sensitive data. This concern dramatically increases users' anxiety and reduces the adoptability of cloud computing in many fields, such as the financial industry and governmental agencies. This paper focuses on this issue and proposes an intelligent cryptography approach, by which the cloud service operators cannot directly reach partial data. The proposed approach divides the file and separately stores the data in the distributed cloud servers. An alternative approach is designed to determine whether the data packets need a split in order to shorten the operation time. The proposed scheme is entitled *Security-Aware Efficient Distributed Storage* (SA-EDS) model, which is mainly supported by our proposed algorithms, including *Alternative Data Distribution (AD2) Algorithm*, *Secure Efficient Data Distributions (SED2) Algorithm* and *Efficient Data Conflation (EDCon) Algorithm*. Our experimental evaluations have assessed both security and efficiency performances and the experimental results depict that our approach can effectively defend main threats from clouds and requires with an acceptable computation time.

1. Introduction

As one of the significant technologies used in cloud computing, the distributed storage has enabled the mass remote data storage via *Storage-as-a-Service* (STaaS) service model. This cloud service model has broadly become an acceptable approach in big data along with the development of Web services and networks [9,20]. Many cloud vendors have given attractive storage service offerings that provide giant and scalable cloud-based storage spaces for users, such as Amazon, Dropbox, Google Drive, and Microsoft's OneDrive [14,19,28]. However, the security issue caused by the operations on cloud side is

* Corresponding author.
E-mail addresses: liyibing@sdu.edu.cn (Y. Li), kg71231w@pace.edu (K. Gai), longfeiqiu2012@gmail.com (L. Qiu), mqiu@pace.edu (M. Qiu), zhh@henu.edu.cn (H. Zhao).

[1] This work was supported in part by the National Science Foundation under Grands CNS-1457506 and NSF CNS-1359557 (Prof. M. Qiu). This work was also supported by the International Science and Technology Cooperation Program of China under Grant 2014DFR70730.

http://dx.doi.org/10.1016/j.ins.2016.09.005
0020-0255/

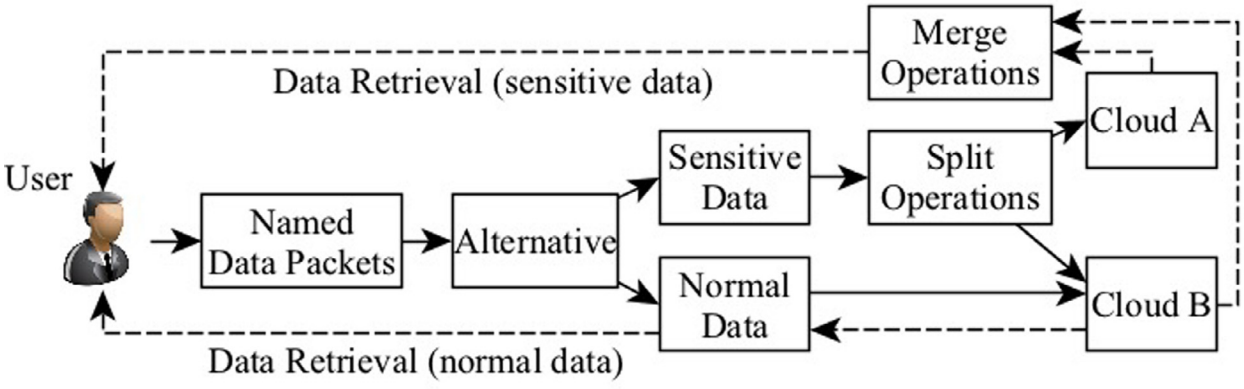

Fig. 1. The architecture of the proposed SA-EDS model.

still an obstacle of using STaaS for enterprises [2,11,13,15,55]. Many cloud users concern about their sensitive data to which the cloud operators have the access [17,54]. This matter embarrasses contemporary implementations of STaaS, even though many prior researches have addressed this field [26,32,40,43].

Moreover, *Mass Distributed Storage* (MDS) has been explored to scale up the data storage size in recent years [23,47]. The high level performances of the scalable computation are considered benefits of implementing MDS. One aspect that needs improvements is to secure distributed data storage [4], in which the threats come from a variety of sides. The distributed storage manner can result in more chances of malicious attacks or abuse activities [5,21], such as attack during data transmissions. Currently, the unexpected operations can also occur at the cloud server side, which are mainly constrained by laws and regulations. Meanwhile, it is difficult to balance functionality and security performances due to cost concerns [50]. Therefore, it is a challenging issue to efficiently secure distributed data in cloud systems, since the risks deriving from different network layers are hardly fully addressed [25,44].

This paper concentrates on the problem of cloud operators abuse issues and attempts to avoid cloud users' data release from cloud servers. We propose an intelligent cryptography approach, named *Security-Aware Efficient Distributed Storage* (SA-EDS) model that is designed to obtain an efficient MDS service, as well as high level security protections. Our proposed mechanism aims to encrypt all data and distributively store the data to the different cloud servers without causing big overheads and latency. Fig. 1 illustrates the architecture of SA-EDS model.

As shown in Fig. 1, user's data are assessed by an alternative process in which searchable named-data-packets techniques are applied. The solid arrow lines represent the data splits and storage operations. The broken arrow lines represent the operational directions of the data retrievals. Normal data will be assigned to a single cloud server. Meanwhile, the data with sensitive information are split into two parts that are assigned to two cloud servers, Cloud A and Cloud B. This process is mainly supported by our proposed algorithm, *Alternative Data Distribution* (AD2) algorithm. Moreover, splitting data process is accomplished by the main algorithm, *Secure Efficient Data Distributions (SED2) Algorithm*, which is designed to spilt data in order to prevent sensitive information from leaking on the cloud side using minimum costs. The sensitive data retrieval needs a decryption process that is supported by our proposed algorithm, *Efficient Data Conflation* (EDCon) algorithm.

The significance of the proposed mechanism is that we provide an adaptable approach for those enterprises that intend to use STaaS but require a high level data storage security, such as the financial service industry. The main problem solved by our proposed scheme is preventing cloud providers from directly reaching users' original data. The main contributions of this paper are twofold:

- We propose a novel cryptography approach for delivering mass distributed storage by which users' original data cannot be directly reached by cloud operators. The proposed method is an effectual cryptography means for defending malicious activities occurred on the cloud server.
- We propose an efficient data split mechanism that does not produce big overheads, as well as ensures data retrievability.

The remainder of this paper follows the structure given below. Recent related work is reviewed and summarized in Section 2. In addition, we represent a motivational example to exemplify the execution process in Section 3. Furthermore, the proposed model and the key concepts used in the model are given in Section 4. Next, Section 5 interprets the main algorithms by pseudo codes and algorithm descriptions. Moreover, we evaluate our proposed model in Section 6 via experimental demonstrations. Finally, Section 7 gives our conclusions.

2. Related work

This section reviews recent research achievements in cloud security issues, which supports the representation of our research background and the theoretical foundation. We addressed two aspects, including current security issues in cloud

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

computing and main active techniques of distributed cloud storage, as well as its challenges. The first aspect explains the main security threats in cloud data storage. The other aspect shows the limitations of current data storage techniques, which also proves the demand of our proposed research.

2.1. Security issues in cloud storage

Security issues have penetrated into most layers of cloud computing, from networks to system managements [46]. Many security issues in networks and data storage are also applicable to cloud computing due to the interconnections between technical applications, such as using *Virtual Machine* (VM). Prior researches explored the security problems and solutions in multiple perspectives.

First, the data management security is an aspect of securing data in cloud computing, which often focus on encryption preparations or data classifications for the purpose of the security [34,39]. Some approaches have been developed to ensure the secure query processing for *Resource Description Framework* (RDF), such as using *eXtensible Access Control Markup Language* (XACML) management policy [8]. Moreover, a selective data encryption is considered a way of reducing computing cost while protecting data in clouds. For example, classifying data in diverse ranks using searchable encryption is an approach for users to alter whether the data need to be encrypted [7,22]. However, most current data management methods assume that the cloud operators do not abuse the data or have limited access to the data. There is a possibility of retrieving information even though the data are encrypted on the cloud side, in some situations.

Next, monitoring and protecting data storage is another dimension in securing cloud data, which considers the data processing or operations occurred in the clouds. It implies that the cloud operators' behaviors are examined or inspected. One of the approaches is using *Attributed-Based Encryption* (ABE) to secure the privacy information when the data are shared among multiple clouds [31,42]. However, restricting cloud operators' access scale can also result in other problems, such as data integration and data intactness [18,38]. Risks of data damage or operation failure rate will be increased if the cloud service providers are fully blocked [11,24,48].

Therefore, from the perspective of data storage, the contradictions between the privacy protection and data processing are difficult to be solved. Most current solutions are trying the balance the trade-off of these two aspects for reducing the total system costs, even though finding out a solution to fitting in most storage scenarios is hard [35].

2.2. Mass Distributed Storage (MDS)

As one of the main techniques used in cloud computing, MDS is used to deal with big data storage in cloud systems [56]. Besides the security issues, a few concerns of using this technique include storage availability, reliability, and accessibility. System integrations become complicated when the size of the data turns into great [10,30]. The concerns of using MSD are usually aligned with security affairs [32,57]. In the particular perspective of cloud-based MDS, a number of main obstacles and explorations are summarized as follows:

First, data synchronizations are facing a great challenge due to the restrictions of computing resources. It is reasonable to receive an efficient synchronizations for a smaller sized users when deploying big data. However, the computing resources face dramatical pressure when the amount of big data users is massive. Recent researches have addressed this problem. For example, an approach [45] was proposed to leverage the notion of local synchronization into asynchronous spiking neural P systems. This approach was designed to optimize the computation power for distributed parallel computing devices.

Moreover, considering the applications in practice, many prior researches explored the implementations of cloud big data storage techniques for improving business processes. One attempt was tracing production processes by using mobile and agent-based computing [12]. Some other researches focuses on the information protections, such as access control mechanisms and trust management [53]. For instance, an approach was proposed to secure instant community data access using trust level classification methodologies [52]. This scheme was effective when users identified the trust communication configurations for *Instant Social Networking* (ISN), which can be supported by the reputation assessments [51]. Another recent research proposed an intercrossed access control scheme for securing multimedia big dat in cloud computing, which used ontology-based authentication classifications [33]. However, these researches mainly focused on securing data transmissions and authentications. The approaches do not have much control when data are stored on the cloud-side.

In addition, it is desired to protect data on cloud servers by using encryption-oriented approaches. Previous researches have also addressed this field, such as *Fully Homomorphic Encryption* (FHE) [41] and ABE. Despite this type of secure mechanisms can effectively protect data from the target attackers, such as external malicious actions and internal improper operations; nonetheless, the efficiency of the data processing can be negative impacted due to the additional computations [3,16]. Some operations cannot even accomplished because of the technical obstacles, such as noises in FHE [6].

In summary, most current active approaches solving the data abuse on cloud-side have two alternatives. The first common approach is using regulatory compliance mechanisms to restrict employees' behaviors [27,36]. This type of paradigm is not well controlled by technical methods. The other method is preventing data from information leakage by encryptions, such as FHE and ABE. But this type of data security cannot satisfy most current industrial demands due to the lowered operation efficiency level and unsolved problems inside the solutions. Our proposed scheme is an attempt that is designed for big data-related applications in which required a higher-level security. A formed distributed storage manner can enable the data to be secured in cloud severs, even though the cloud operators have the access to the data.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

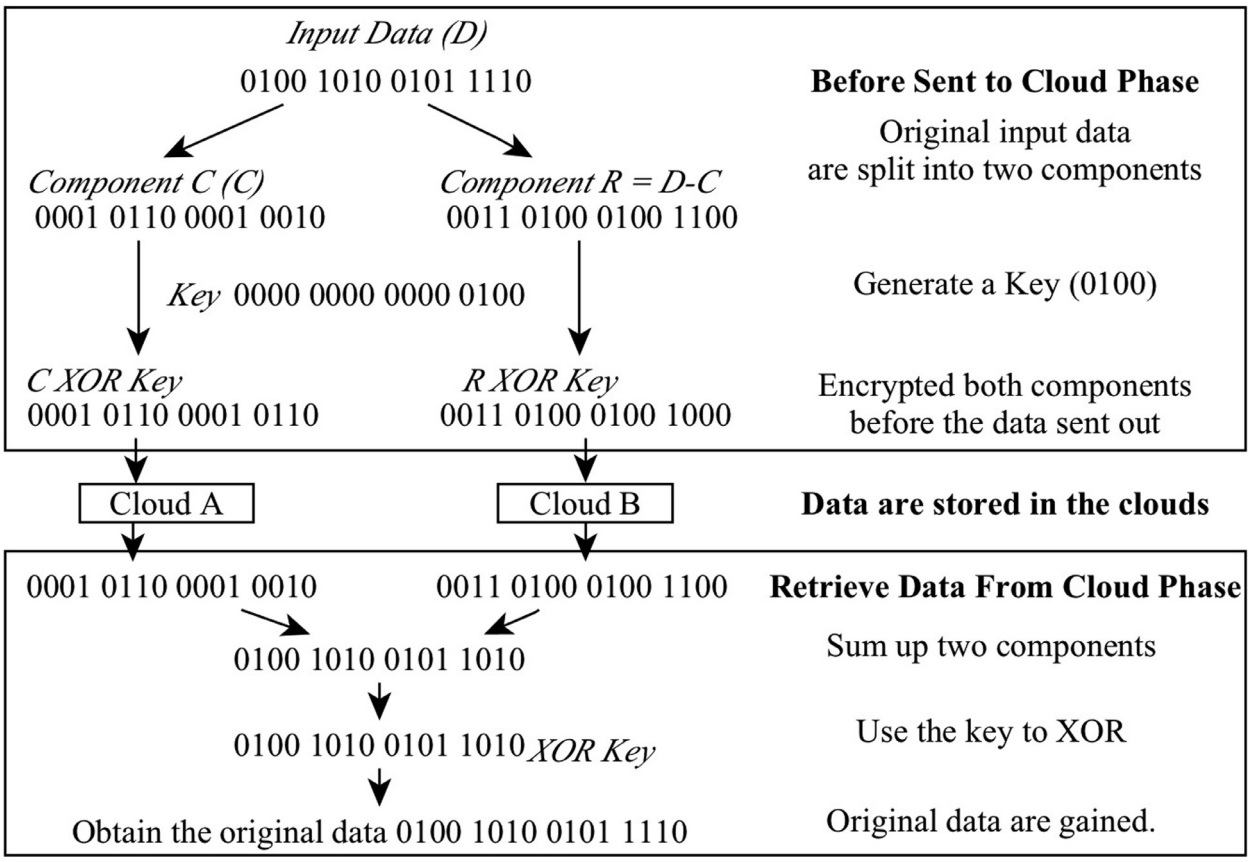

Fig. 2. Motivational example: the main phases of the given example using the proposed scheme.

3. Motivational example

In this section, a motivation example explains the crucial part of the proposed model, which is securing data packets with sensitive information. The process consists of splitting data packets and data packets retrievals. This scenario takes place in the financial industry, in which users' sensitive information needs to be highly protected on the cloud side. Assume that there are two remote cloud storage servers, *A* and *B*. There is an input data *D* that is *0100 1010 0101 1110*. Our goal is to distributively store data D to cloud server A and B and ensure cloud operators at A and B cannot directly touch the data. The storage process also needs to satisfy both high security and low latency requirements. Fig. 2 illustrates the main procedures of implementing our model.

The model has two main phases, which are *Before Sent to Cloud Phase* and *Retrieve Data From Cloud Phase*. As shown in Fig. 2, before the data were sent to remote cloud servers, we spilt the data into two components, C and R, represented as $D \rightarrow \langle C, R \rangle$. We randomly generate the component C as *0001 0110 0001 0010* so that the other component R can be gained by *D-C = 0011 0100 0100 1100*. The original input data D = C + R. Next, randomly generate a Key valued as *0100* and use the key to do XOR operators with both C and R. The process can be represented as α=*C XOR key* and β = *R XOR Key*. The values of α and β are *0001 0110 0001 0110* and *0011 0100 0100 1000*. After this calculation, we send out these two data to Cloud A and B, separately.

Using this approach can perfectly spilt the original data into two parts, from which the information carried by the data cannot be obtained on two cloud servers. From the perspective of attackers or abuse operators, it is almost impossible to guess both data components and key values.

Moreover, when user needs to retrieve their data from cloud servers, the service request can be accomplished by finishing *Retrieve Data From Cloud Phase*, as shown in Fig. 2. There are a few steps to finish this phase. First, gain data from both Cloud A and B and use the key to do XOR operations to them. We obtain two data from this step, which are *0001 0110 0001 0010* and *0011 0100 0100 1100*. Next, we sum up these two data and obtain *0100 1010 0101 1010*. Then do a XOR operation to this data and obtain the original data *0100 1010 0101 1110*.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

This scheme can effectively protect users' data, since the *Key* value is randomly generated and any split data do not carry any content information. Adversaries cannot gain sensitive information even though they touch the data. Section 4 defines the threat model and describes the procedure of the proposed model.

4. Concepts and the proposed model

4.1. Problem formulations

We implement our model aligning with the architecture shown in Fig. 1. The input data are partitioned into functional units before they are stored. Our approach is designed to divide the sensitive data into two encrypted parts for distributed cloud storage. The main problem addressed by SA-EDS is to avoid cloud providers' employees from reaching data without reducing the efficiency performance. We define the main problem addressed as a *Security Improvement Problem in Distributed Storage* (SIPDS) that is described as follows:

Definition 1Security Improvement Problem in Distributed Storage (SIPDS)**.** Given the initial input data and storage cloud servers. The target problem is to find out a solution that can successfully store in the cloud servers and ensure the data cannot be reached by cloud operators without greatly increasing execution time.

The inputs include the initial data that consist of a string of data packets with sensitive information. The outputs are two separate data packets that will be transmitted to different cloud storage servers. The new generated data packets need to perfectly hide the sensitive information so that the cloud operators cannot read the information, even though they have the access to the data. A low level execution time of the data conversion is required.

4.2. Security-Aware Efficient Distributed Storage (SA-EDS) model

Our proposed SA-EDS model mainly consists of two components, namely *Deterministic Process* (DP) and *Data Distributed Storage Process* (D2SP). The first component is designed to determine whether the input data packets demand a high level security guarantee. The other component is used to protect data from the unexpected activities that are caused by cloud-side employees. It is a core part of our proposed model. The subsections below provide detailed explanations about these two components.

4.2.1. Deterministic Process (DP)

The DP operations determine whether the data packets will be distributively stored in different cloud servers. The distributed data storage will be applied to those data containing sensitive information. Our proposed model only executes XOR operations. The whole scheme to operate DP is as follows:

- *Setup* The input data are searchable named-data-packets that are alternative friendly, by which the security level can be determined by the named label. The search named-data-packets mean the data packets are named by labels for the purpose of search. Data owners or cloud service providers configure a pool consisting of the name labels that are associated to those data packets with a high security requirement.
- *Deterministic Alternative* When the named-data-packet contains a name label that belongs to the PNL, the proposed model carry out D2SP operations. Otherwise, the model operates XOR to the data packets and sends the encrypted data packets to the cloud servers.
- *Data Retrieval* Two types of data retrieval are required in our model. The first type is designed for the sensitive data retrieval, which uses EDCon algorithm (Algorithm 3). The other type is for normal data retrieval, which requires an XOR operation to the data packets.

The purpose of conducting DP in the proposed model is to reduce the costs of the computing resource and computation workloads.

4.2.2. Data Distributed Storage Process (D2SP)

Fig. 3 represents a high level workflow structure of D2SP in the proposed SA-EDS model. The figure illustrates the principle mechanism of our paradigm. As mentioned in Section 3, there are mainly two phases in the model, which are aligned with the middle and the left boxes in the figure. These two phases form two crucial procedures during the data transmissions. One is processing data in order to partition the input data strings into two separate data strings. The other one is used to merge the data to obtain the original data.

At the *Before Sent to Cloud Phase*, we partition the input data D into two separate components. As shown in the figure, there are two encrypted components, A and B. This process is accomplished in a few steps. First, we generate a random parameter data C and use it to produce new data packets by operating D-C. Second, we use a random key held by users to operate XOR to both C and R. This key needs to stored in a special register at user side. Finally, both encrypted data packets were sent to separate cloud servers.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

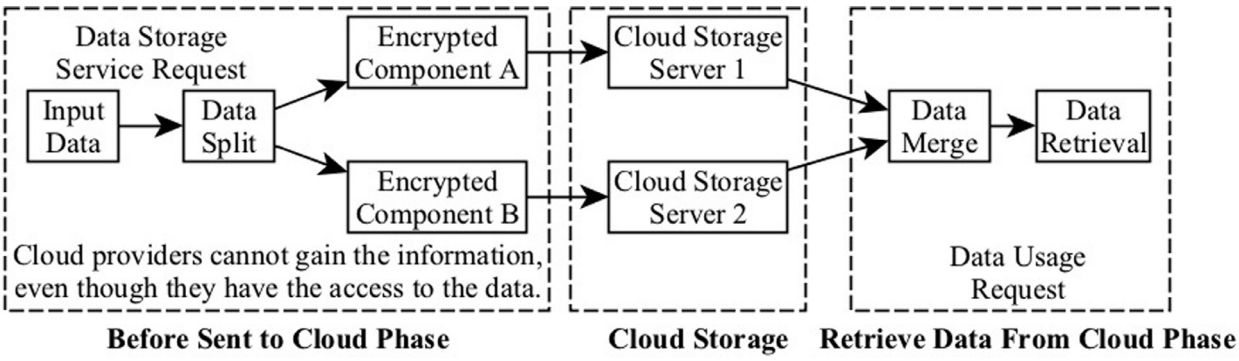

Fig. 3. High level workflow structure of splitting data packets in the Data Distributed Storage Process (D2SP) within SA-EDS model.

Furthermore, at the *Retrieve Data From Cloud Phase*, data users need to receive data packets from both cloud providers. Attaining the original data needs a series of operations after the data packets are received from cloud sides. First, the corresponding data packets need to be summed up to produce the new data string. Next, users will use the key to do two actions followed by the below order, XOR operation to the new data string and add the Key data value after. The original data will be gained after this procedure is finished.

4.3. Threat models

The cloud server usually plays a trustable role in cloud service deployment models, such that many cloud service model designers assume that the operators on the cloud-side are secure. However, many risks are caused by the unexpected behaviors made by cloud operators rather than the malicious attacks. In many situations, it turns into a morality issue instead of a technical problem, since cloud employees usually need the access to the data for the purpose of the data governance even though the activities are restricted by the regulations. Meanwhile, data are not secure although encryptions are applied. The information can be released in a great chance when the malicious operations are given. Therefore, we consider that the main threats came from cloud operators and define two threat models, based on the current cloud practices [7].

1. *Anti-Regulatory Compliance Threat (ARCT) Model* We assume that the cloud-side employees have the intention to access to the data without following the regulations in this model. Cloud employees have the access to the server and know the *key* to the data encryption.
2. *Malicious Access Threat (MAT) Model Knowing Information Background* We assume that the cloud-side operators intend to have the malicious access to the data and gain the information in this model. Cloud operators have the background knowledge about the data stored in the cloud server. The operators can have a guess about the information even though the data are encrypted, such that the information can be retrieved when the encryption security level is not high.

These two threat models can be formulated by the following Definition 2.

Definition 2Anti-Regulatory Compliance Threat Model:. $\exists$ a key K to decrypt a data packet D on the cloud, as $K \rightarrow D$. Assume that cloud operators use K to access to D without permissions from the data owner.

4.4. Design goals

Our proposed system aims to simultaneously achieve a few targeted performances as follows, which can guarantee the data security required by particular data users, such as financial practitioners or auditing professionals:

- *Preventing threats from internal threats:* We aim to achieve a higher-level security data storage splitting data into diverse cloud servers, in which internal threats can neither abuse the data nor retrieve the information from the stored data on the server.
- *Data protection against external threats:* The proposed system will protect data from the attacks issued by the external adversaries. Data need to be encrypted during the transmission process.
- *High efficiency data processing:* Our system will also avoid high communication and computation overhead in order to lower down the latency.

Our experiments evaluated these design goals in Section 6. The succeeding section describes the main algorithms used in our proposed model.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

5. Algorithms

In this section, we give descriptions of our proposed algorithms. Three main algorithms support our security model, which includes *Alternative Data Distribution* (AD2), *Secure Efficient Data Distributions* (SED2) and *Efficient Data Conflation* (EDCon) algorithms. The sections below explain the detailed mechanism of the algorithms, respectively.

5.1. Alternative Data Distribution (AD2) algorithm

AD2 algorithm is designed to determine whether the data packet needs to be split and stored in distributed cloud servers. AD2 is a parallelizable algorithm that can be compatible with big data framework, such as Hadoop MapReduce. For example, the data packets with sensitive information can be distinguished by using fuzzy keyword searching techniques. The mechanism of splitting data is grouping the named-data-packets by the name labels. The inputs include the *Named-Data-Packets* (NDP), $\{D(N)\}$, and the *Pre-stored Name List* (PNL). We define the PBL as the configured list that shows those data packets that contain sensitive information and are required to be highly protected. The NDP needs a split operation if its name is included in the list. The output of this algorithm is a *Data Packet* (D) that will execute the data segregation.

For the NDP, the data can be either plain texts or cipher texts but the requirement is the data are searchable by using named label. Prior researches [29,49] had provided a few options for searchable encryptions such that the named labels can be used as the deterministic references for splitting data. Each NDP has one or a few name labels $\{L_1, L_2, \ldots, L_n\}$. We configure that PNL consists of a set of searchable labels, $\{N_1, N_2, \ldots, N_n\}$, which is associated with those data packets containing sensitive information.

Pseudo codes of AD2 algorithm are shown in Algorithm 1. We describe the main steps of Algorithm 1 in the following statement:

1. Input the searchable named-data-packets that are searchable and PNL.
2. For all named-data-packets, we search each data packet and see whether there is a name label that matches searchable labels in PNL.
3. If a match is found, execute SED2 algorithm by which the data packets are split into two parts and stored in distributed cloud servers. In this process, the split data packets include α and β.
4. Otherwise, execute an XOR operation to the data packet and generate an encrypted data packet D_{xor}.
5. Output the encrypted data packets, including D_{xor}, α, and β.

Algorithm 1 Alternative Data Distribution (AD2) algorithm.

Require: *NDP*, *PNL*
Ensure: D_{xor}, α, β

```
Input NDP, PNL
for ∀NDP do
  for each data packet do
    if ∃ a L_i ∈ PNL then
      Execute SED2 Algorithm /* Algorithm 2 */
      Generate α and β
    else
      Do XOR operation to the data packet
      /*Do XOR operation before the data packet is sent out*/
      Generate D_xor
    end if
  end for
  Obtain the values of D
end for
Output D_xor
```

5.2. Secure Efficient Data Distributions (SED2) algorithm

SED2 Algorithm is designed to accomplish the data processing before they are transmitted to the cloud side. This algorithm is aligned with the procedure *Before Sent to Cloud Phase* in Fig. 3. The inputs of this algorithm include the *Data Packet* (D), a random split binary parameter *C*. The outputs include two separate encrypted data α and β.

Executing SED2 algorithm can competently defend the threat models mentioned in Section 4.3. In ARCT threat model, assume that cloud employees have the *Key* and can access to the data on the server. This condition is not sufficient for cloud employees to obtain information from the data, since all that the employees obtain is the partial data. The other

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

partial data are stored in some other places where the cloud employees do not have the access. Partial data do not contain any information since the original data will not be obtained until two parts are operated together. Moreover, in MAT threat model, the adversaries have background information about the data and intend to abuse the data. However, the adversaries will encounter the same problem as attackers in ARCT model. No information will be released since adversaries can only steel data from the server to which they have the access. Therefore, our proposed scheme can effectively defend both threat models in the theoretical perspective. Pseudo codes of SED2 algorithm is given in Algorithm 2.

Algorithm 2 Secure Efficient Data Distributions (SED2) algorithm.

Require: D, C
Ensure: α, β

Input D, C
Initialize $R \leftarrow 0$, $\alpha \leftarrow 0$, $\beta \leftarrow 0$
/* C is a random binary that is shorter than D */
Randomly generate a key K
for ∀ input data packets **do**
  if $D \neq C$ && $C \neq 0$ **then**
    DO $R \leftarrow D - C$
    $\alpha \leftarrow C \bigoplus K$
    $\beta \leftarrow R \bigoplus K$
  end if
end for
Output α, β

The main steps of Algorithm 2 are given as follows:

1. Input data packet D and C. Data C needs to be a non-empty set that is shorter than D. C should not be as same as D. Create and initialize a few dataset, R, α, and β; assign 0 value to each of them.
2. Randomly generate a key K that is stored at the user's special register for the purpose of encryption and decryption. This is the crucial part for protecting privacy before the data are sent out.
3. We calculate the value of R by (D-C), then execute two XOR operations to obtain the data value stored in the clouds. The data in the remote storage are denoted to α and β. We use the following formulas to obtain α and β: $\alpha = C \bigoplus K$; $\beta = R \bigoplus K$.
4. Output α and β and separately store them in the different cloud servers.

5.3. Efficient Data Conflation (EDCon) algorithm

EDCon algorithm is designed to enable users to obtain the information by converging two data components from distributed cloud servers. The corresponding phase shown in Fig. 3 is *Retrieve Data From Cloud Phase*. Inputs of this algorithm include two data components from cloud servers, α, β, and K. Output is user's original data D. Algorithm 3 shows the pseudo codes of SED2.

Algorithm 3 Efficient Data Conflation (EDCon) algorithm.

Require: α, β, K
Ensure: D

Input α, β, K
Initialize $\gamma \leftarrow 0$, $\gamma' \leftarrow 0$, $D \leftarrow 0$
/* User receives α, β from separate cloud servers*/
$\gamma \leftarrow \alpha \bigoplus K$
$\gamma' \leftarrow \beta \bigoplus K$
$D \leftarrow \gamma + \gamma'$
Output D

Main phases of Algorithm 3 are explained as follows:

1. Input the data, α and β, that are acquired from different cloud servers. The user obtains the key K from the special register. Initialize a few dataset γ, γ', and D for the operation needs.
2. Execute the XOR operation to both α and β by using K. Assign the value to γ and γ', respectively. $\gamma \leftarrow \alpha \bigoplus K$, $\gamma' \leftarrow \alpha \bigoplus K$
3. Sum up γ and γ' and assign the summation to D, as $D = \gamma + \gamma'$.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

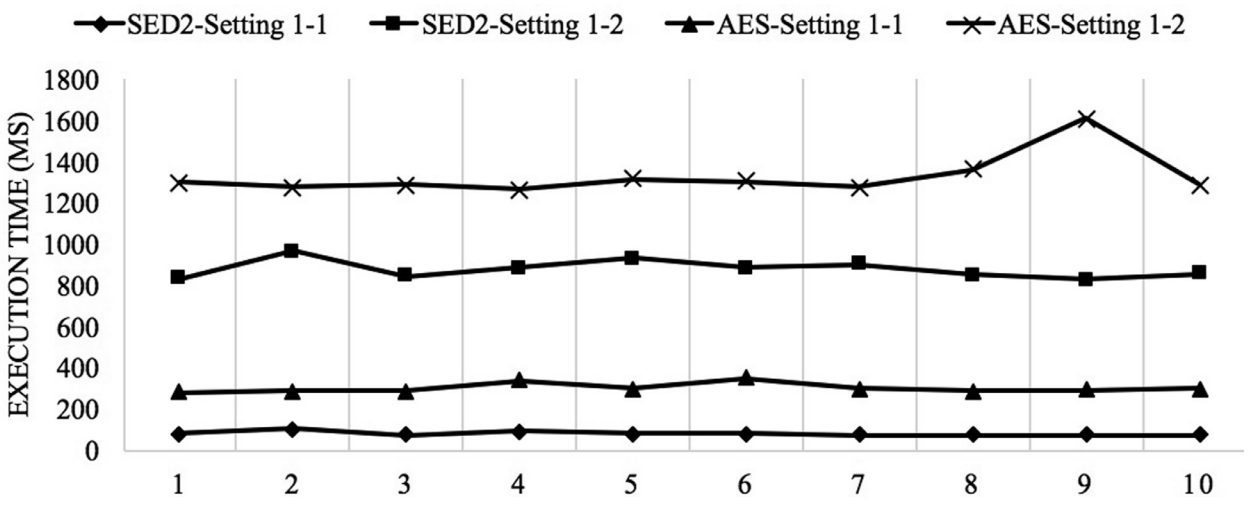

Fig. 4. Comparison on the execution time between EDS and AES under setting 1-1 and 1–2.

4. Output D that is the original data.

The next section represents our experimental evaluations and the results.

6. Experiment and the results

We represented our experimental configurations and partial experimental results in this section. The experimental design focused on the adoption of the proposed model in the perspective of the execution time. The evaluations also considered the comparisons with current active cryptography method.

6.1. Experimental configuration

There were two assessment dimensions in our experiment. The first dimensions was evaluating execution time differences between EDS and *Advanced Encryption Standard* (AES) [1,37] that is a broadly accepted symmetric encryption algorithm. we demonstrated that our proposed scheme had an advantage in reducing execution time. The other dimension was assessing whether the execution time of the proposed scheme was impacted on the data size, which was considered an important aspect in wireless communications.

The experimental environment was configured as following. We simulated the cloud environment via our simulator that was designed and developed for the experiment. The hardware setting used in our experiment was an HP server with 8-core CPU, 8 GB memory, and Mango DB. We installed a VMWare workstation and a Ubuntu 15.04 LTS Server on the side of the VMWare workstation.

In order to evaluate the expected performance dimensions, we evaluated the proposed model by assessing its execution time while different input data sizes were operated. Based on this configuration, we showed partial experimental settings as follows:

1. Setting 1: evaluations based on the data required to be encrypted. The assessment was processed by different input data sizes, as follows: Setting 1-1: 1 KB, Setting 1–2: 1 MB, Setting 1–3: 10 MB, Setting 1–4: 50 MB, Setting 1–5: 250 MB, Setting 1–6: 500 MB.
2. Setting 2: evaluations based on the data retrieved from cloud servers. The assessment was processed by different retrieval data sizes, as follows: Setting 2-1: 1 KB, Setting 2-2: 1 MB, Setting 2–3: 10 MB, Setting 2–4: 50 MB, Setting 2–5: 250 MB, Setting 2–6: 500 MB.

6.2. Experimental results

This section displayed a few experimental results made in our performance evaluations.

Figs. 4 and 5 illustrated a comparison of the execution time between EDS and AES. We used the same sized input data and examined the encryption time consumptions. The figure showed some results that were generated under setting 1-1 and 1–2. According to the lines shown in Figs. 4 and 5, our proposed scheme had a shorter execution time than AES under both displayed settings. The decryption time required a longer time period under both settings.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

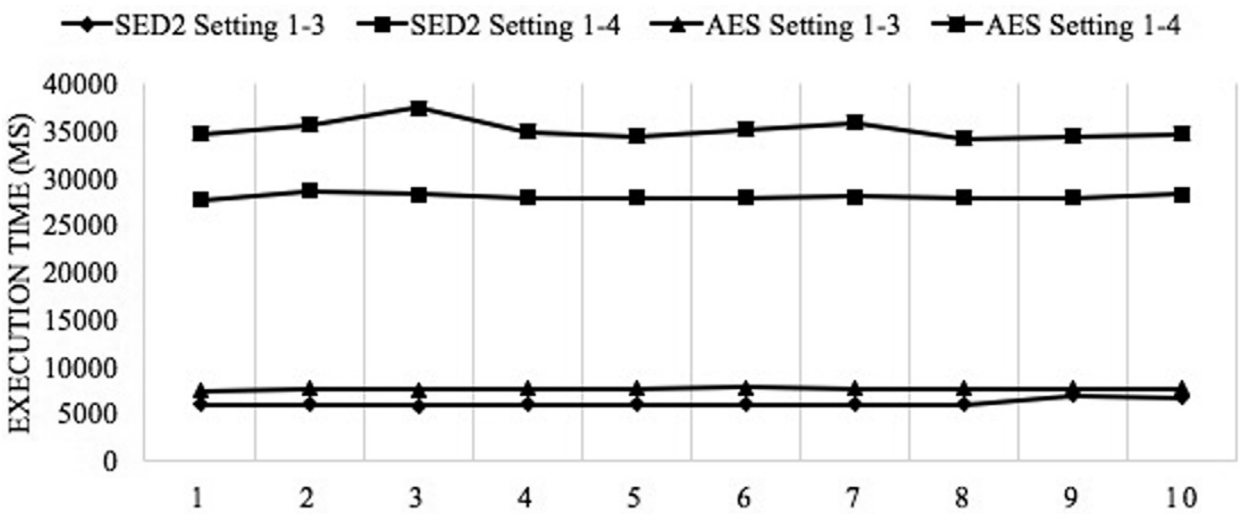

Fig. 5. Comparison on the execution time between EDS and AES under setting 1–3 and 1–4.

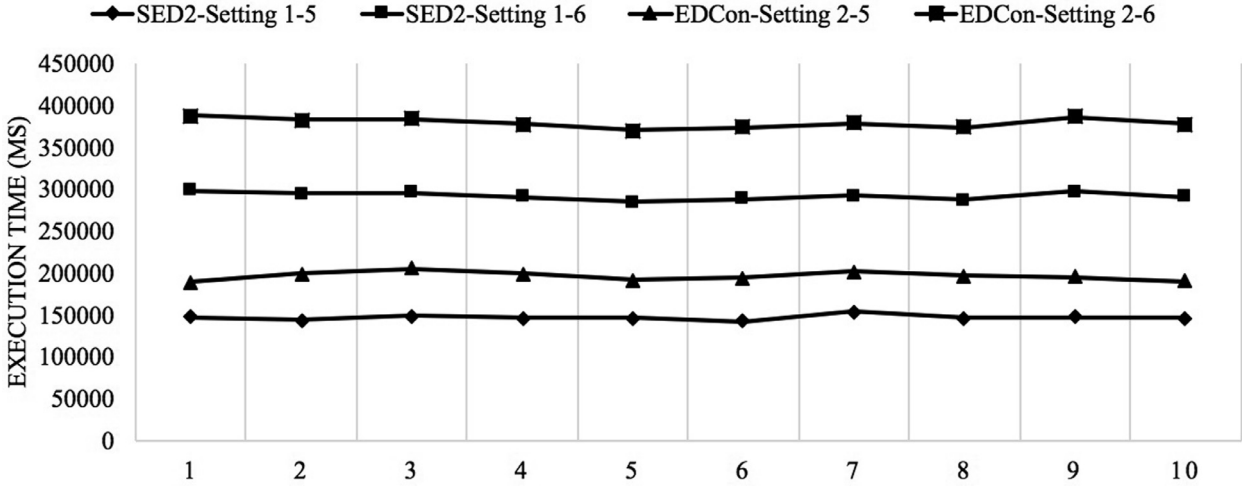

Fig. 6. Comparisons between data sent out and data retrieval under Setting 5 and 6.

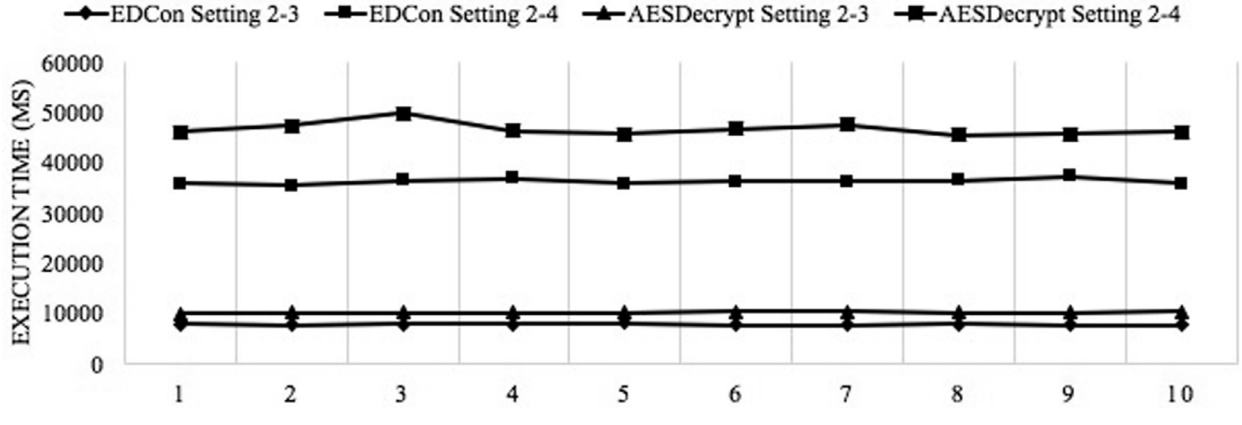

Fig. 7. Comparisons between EDCon and AES decryption under setting 2–3 and 2–4.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

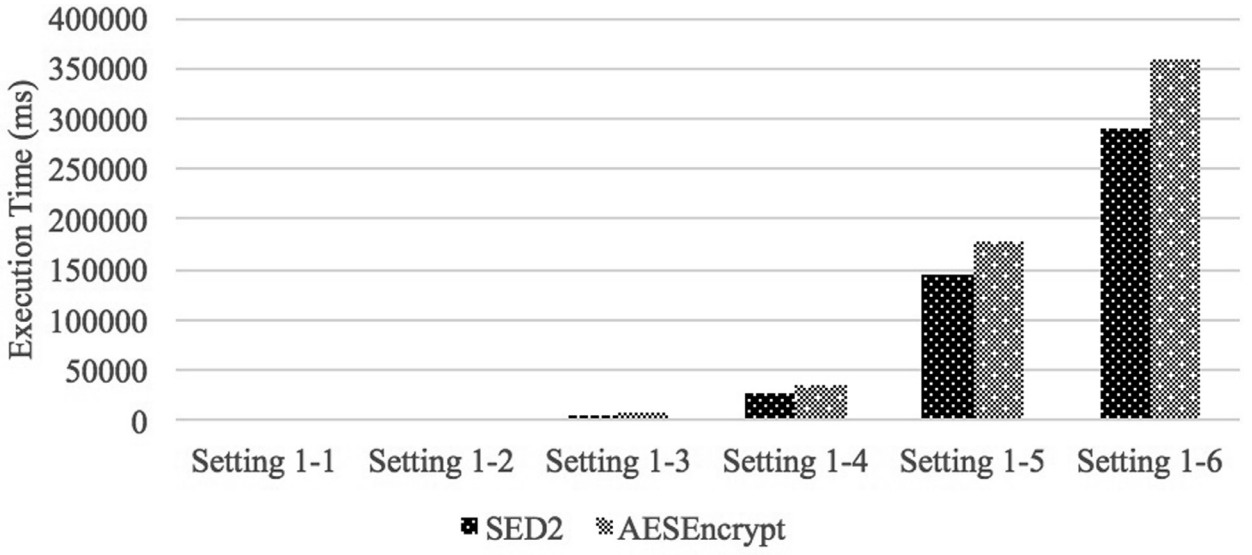

Fig. 8. Comparisons of the encryption execution time between SED2 (before document is sent out) and AES using settings 1-1, 1–2, 1–3, 1–4, 1–5, and 1–6.

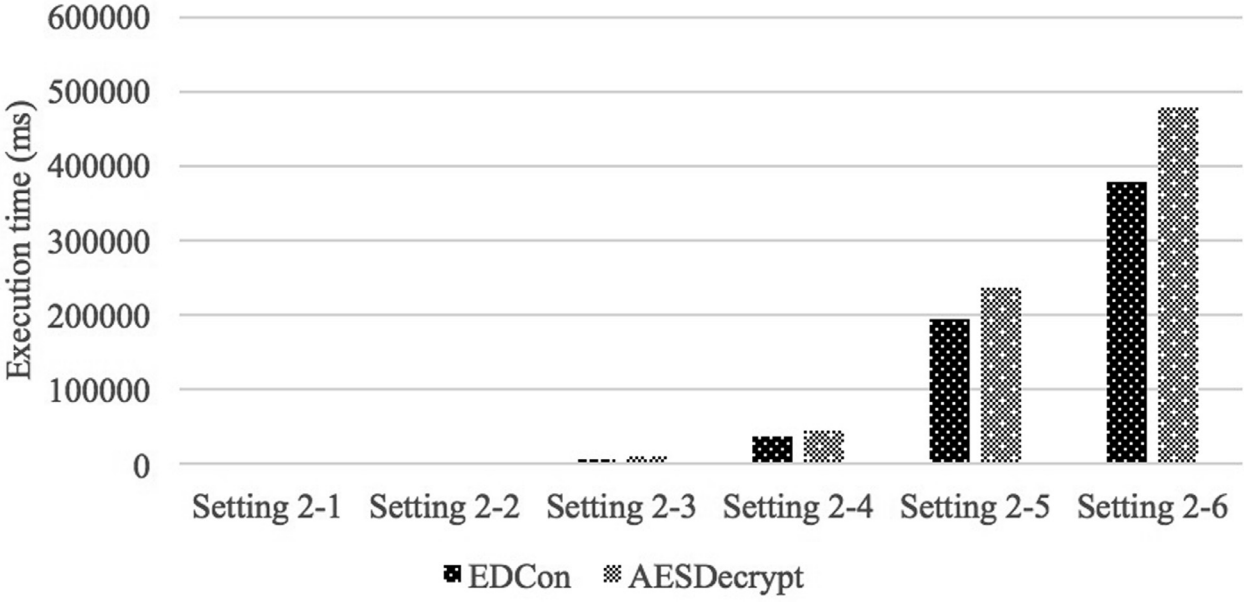

Fig. 9. Comparisons of the data retrieval execution time between EDCon and AES using settings 2-1, 2-2, 2–3, 2–4, 2–5, and 2–6. Document is gained from the cloud server.

Meanwhile, we also assessed the calculating performance differences given by encryptions and decryptions. Fig. 6 represented execution time differences between the encryption and decryption when the data sizes were varied. The horizontal axis represents the amount of the evaluations. The figure showed that the data that needed decryptions were impacted by the data size. The execution time became longer when the data size increased.

Furthermore, Fig. 7 illustrates comparisons of the execution time between EDCon and AES decryption under setting 2–3 and 2–4. The execution time length of our proposed approach is slightly longer than AES.

Moreover, Fig. 8 represented the encryption execution time differences for both SED2 and AES while the data sizes were varied. The experimental evaluations were under settings 1-1, 1–2, 1–3, 1–4, 1–5, and 1–6. We simulated the data encryption processes before the data were sent to cloud-side servers. As displayed in the figure, the encryption execution time consumptions were were associated with the data sizes. Our proposed scheme consumed less computation time than AES.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

In addition, Fig. 9 showed a comparison of the data retrieval processing time while the data sizes were different. The decryption time consumptions had a similar situation to the data encryption. The data processing time had a positive relationship with the data sizes. Our proposed approach needed a shorter processing time than AES when the examined settings were applied.

7. Conclusions

This paper focused on the problem of the cloud data storage and aimed to provide an approach that could avoid the cloud operators reaching user' sensitive data. Addressing this goal, we proposed a novel approach entitled as *Security-Aware Efficient Distributed Storage* (SA-EDS) model. In this model, we used our proposed algorithms, including *Alternative Data Distribution* (AD2), *Secure Efficient Data Distributions* (SED2) and *Efficient Data Conflation* (EDCon) algorithms. Our experimental evaluations had proved that our proposed scheme could effectively defend major threats from cloud-side. The computation time was shorter than current active approaches. Future work would address securing data duplications in order to increase the level of data availability since any of datacenter's down will cause the failure of data retrievals.

References

[1] A. Alahmadi, M. Abdelhakim, J. Ren, T. Li, Defense against primary user emulation attacks in cognitive radio networks using advanced encryption standard, IEEE Trans. Inf. Forensics Secur. 9 (5) (2014) 772–781.
[2] M. Ali, S. Khan, A. Vasilakos, Security in cloud computing: Opportunities and challenges, Inf. Sci. 305 (2015) 357–383.
[3] R. Aliev, W. Pedrycz, B. Fazlollahi, O. Huseynov, A. Alizadeh, B. Guirimov, Fuzzy logic-based generalized decision theory with imperfect information, Inf. Sci. 189 (2012) 18–42.
[4] G. Ateniese, K. Fu, M. Green, S. Hohenberger, Improved proxy re-encryption schemes with applications to secure distributed storage, ACM Trans. Inf. Syst. Secur. 9 (1) (2006) 1–30.
[5] J. Baek, Q. Vu, K. Liu, X. Huang, Y. Xiang, A secure cloud computing based framework for big data information management of smart grid, IEEE Trans. Cloud Comput. 3 (2) (2015) 233–244.
[6] Z. Brakerski, C. Gentry, V. Vaikuntanathan, (Leveled) fully homomorphic encryption without bootstrapping, ACM Trans. Comput. Theory 6 (3) (2014) 13.
[7] N. Cao, C. Wang, M. Li, K. Ren, W. Lou, Privacy-preserving multi-keyword ranked search over encrypted cloud data, IEEE Trans. Parallel Distrib. Syst. 25 (1) (2014) 222–233.
[8] D. Chadwick, K. Fatema, A privacy preserving authorisation system for the cloud, J. Comput. Syst. Sci. 78 (5) (2012) 1359–1373.
[9] V. Chang, M. Ramachandran, Towards achieving data security with the cloud computing adoption framework, IEEE Trans. Serv. Comput. 9 (1) (2016) 138–151.
[10] C. Chen, M. Won, R. Stoleru, G. Xie, Energy-efficient fault-tolerant data storage and processing in mobile cloud, IEEE Trans. cloud comput. 3 (1) (2015) 28–41.
[11] C. Chen, C. Zhang, Data-intensive applications, challenges, techniques and technologies: A survey on big data, Inf. Sci. 275 (2014) 314–347.
[12] M. Cimino, F. Marcelloni, Autonomic tracing of production processes with mobile and agent-based computing, Inf. Sci. 181 (5) (2011) 935–953.
[13] K. Costa, L. Pereira, R. Nakamura, C. Pereira, J. Papa, A. Falcão, A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks, Inf. Sci. 294 (2015) 95–108.
[14] L. Darrell, Unlimited cloud storage at amazon.com, inc on black friday, Url=http://www.bidnessetc.com/58232-unlimited-cloud-storage-at-amazoncom-inc-on-black-friday/.
[15] Y. Ding, Y. Hu, K. Hao, L. Cheng, MPSICA: An intelligent routing recovery scheme for heterogeneous wireless sensor networks, Inf. Sci. 308 (2015) 49–60.
[16] K. Gai, Z. Du, M. Qiu, H. Zhao, Efficiency-aware workload optimizations of heterogenous cloud computing for capacity planning in financial industry, in: The 2nd IEEE International Conference on Cyber Security and Cloud Computing, IEEE, New York, USA, 2015, pp. 1–6.
[17] K. Gai, S. Li, Towards cloud computing: a literature review on cloud computing and its development trends, in: 4th International Conference on Multimedia Information Networking and Security, Nanjing, China, 2012, pp. 142–146.
[18] K. Gai, L. Qiu, M. Chen, H. Zhao, M. Qiu, SA-EAST: Security-aware efficient data transmission for ITS in mobile heterogeneous cloud computing, ACM Trans. Embedded Comput. Syst. 1 (2016) 99.
[19] K. Gai, L. Qiu, H. Zhao, M. Qiu, Cost-aware multimedia data allocation for heterogeneous memory using genetic algorithm in cloud computing, IEEE Trans. Cloud Comput. 1 (2016) 99.
[20] K. Gai, M. Qiu, L. Chen, M. Liu, Electronic health record error prevention approach using ontology in big data, in: 17th IEEE International Conference on High Performance Computing and Communications, New York, USA, 2015, pp. 752–757.
[21] K. Gai, M. Qiu, L. Tao, Y. Zhu, Intrusion detection techniques for mobile cloud computing in heterogeneous 5G, Secur. Commun. Netw. (2015) 1–10.
[22] K. Gai, M. Qiu, B. Thuraisingham, L. Tao, Proactive attribute-based secure data schema for mobile cloud in financial industry, in: The IEEE International Symposium on Big Data Security on Cloud, IEEE 17th International Conference on High Performance Computing and Communications, New York, USA, 2015, pp. 1332–1337.
[23] K. Gai, M. Qiu, H. Zhao, Security-aware efficient mass distributed storage approach for cloud systems in big data, in: 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), IEEE, New York, USA, 2016, pp. 140–145.
[24] K. Gai, M. Qiu, H. Zhao, W. Dai, Anti-counterfeit schema using monte carlo simulation for e-commerce in cloud systems, in: The 2nd IEEE International Conference on Cyber Security and Cloud Computing, IEEE, New York, USA, 2015, pp. 74–79.
[25] K. Gai, M. Qiu, H. Zhao, L. Tao, Z. Zong, Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing, J. Netw. Comput. Appl. 59 (2015) 46–54.
[26] K. Gai, M. Qiu, H. Zhao, J. Xiong, Privacy-aware adaptive data encryption strategy of big data in cloud computing, in: The 3rd IEEE International Conference on Cyber Security and Cloud Computing, The 2nd IEEE International Conference of Scalable and Smart Cloud, IEEE, Beijing, China, 2016, pp. 273–278.
[27] E. Herrera-Viedma, F. Cabrerizo, J. Kacprzyk, W. Pedrycz, A review of soft consensus models in a fuzzy environment, Inf. Fusion 17 (2014) 4–13.
[28] D. Howley, Is microsoft's onedrive the best cloud storage service?, Url=https://www.yahoo.com/tech/microsoft-kills-unlimited-onedrive-accounts-175927221.html.
[29] H. Li, D. Liu, Y. Dai, T. Luan, Engineering searchable encryption of mobile cloud networks: When qoe meets qop, IEEE Wireless Commun. 22 (4) (2015) 74–80.
[30] J. Li, M. Qiu, Z. Ming, G. Quan, X. Qin, Z. Gu, Online optimization for scheduling preemptable tasks on iaas cloud systems, J. Parallel Distrib. Comput. 72 (5) (2012) 666–677.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005

[31] M. Li, S. Yu, Y. Zheng, K. Ren, W. Lou, Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption, IEEE Trans. Parallel Distrib. Syst. 24 (1) (2013) 131–143.
[32] Y. Li, W. Dai, Z. Ming, M. Qiu, Privacy protection for preventing data over-collection in smart city, IEEE Trans. Comput. 65 (5) (2016) 1339–1350.
[33] Y. Li, K. Gai, Z. Ming, H. Zhao, M. Qiu, Intercrossed access control for secure financial services on multimedia big data in cloud systems, in: ACM Transactions on Multimedia Computing Communications and Applications, 2016, p. 1.
[34] Q. Liu, G. Wang, J. Wu, Time-based proxy re-encryption scheme for secure data sharing in a cloud environment, Inf. Sci. 258 (2014) 355–370.
[35] S. Liu, Q. Qu, L. Chen, L. Ni, SMC: A practical schema for privacy-preserved data sharing over distributed data streams, IEEE Trans. Big Data 1 (2) (2015) 68–81.
[36] C. Modi, D. Patel, B. Borisaniya, A. Patel, M. Rajarajan, A survey on security issues and solutions at different layers of cloud computing, J. Supercomput. 63 (2) (2013) 561–592.
[37] M. Mozaffari-Kermani, A. Reyhani-Masoleh, A lightweight high-performance fault detection scheme for the advanced encryption standard using composite fields, IEEE Trans. Very Large Scale Integr. Syst. 19 (1) (2011) 85–91.
[38] A. Parakh, S. Kak, Online data storage using implicit security, Inf. Sci. 179 (19) (2009) 3323–3331.
[39] W. Pedrycz, Allocation of information granularity in optimization and decision-making models: Towards building the foundations of granular computing, Eur. J. Oper. Res. 232 (1) (2014) 137–145.
[40] W. Pedrycz, M. Song, A granulation of linguistic information in AHP decision-making problems, Inf. Fusion 17 (2014) 93–101.
[41] T. Plantard, W. Susilo, Z. Zhang, Fully homomorphic encryption using hidden ideal lattice, IEEE Trans. Inf. Forensics Secur. 8 (12) (2013) 2127–2137.
[42] M. Qiu, K. Gai, B. Thuraisingham, L. Tao, H. Zhao, Proactive user-centric secure data scheme using attribute-based semantic access controls for mobile clouds in financial industry, Future Gener. Comput. Syst. (2016) 1.
[43] M. Qiu, L. Zhang, Z. Ming, Z. Chen, X. Qin, L. Yang, Security-aware optimization for ubiquitous computing systems with SEAT graph approach, J. Comput. Syst. Sci. 79 (5) (2013) 518–529.
[44] M. Qiu, M. Zhong, J. Li, K. Gai, Z. Zong, Phase-change memory optimization for green cloud with genetic algorithm, IEEE Trans. Comput. 64 (12) (2015) 3528–3540.
[45] T. Song, L. Pan, G. Păun, Asynchronous spiking neural P systems with local synchronization, Inf. Sci. 219 (2013) 197–207.
[46] C. Wang, S. Chow, Q. Wang, K. Ren, W. Lou, Privacy-preserving public auditing for secure cloud storage, IEEE Trans. Comput. 62 (2) (2013) 362–375.
[47] H. Wang, Z. Xu, H. Fujita, S. Liu, Towards felicitous decision making: An overview on challenges and trends of big data, Inf. Sci. 367 (2016) 747–765.
[48] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, A. Vasilakos, Security and privacy for storage and computation in cloud computing, Inf. Sci. 258 (2014) 371–386.
[49] M. Wen, R. Lu, J. Lei, H. Li, X. Liang, X. Shen, Sesa: An efficient searchable encryption scheme for auction in emerging smart grid marketing, Secur. Commun. Netw. 7 (1) (2014) 234–244.
[50] G. Wu, H. Zhang, M. Qiu, Z. Ming, J. Li, X. Qin, A decentralized approach for mining event correlations in distributed system monitoring, J. Parallel Distrib. Comput. 73 (3) (2013) 330–340.
[51] Z. Yan, Y. Chen, Y. Shen, A practical reputation system for pervasive social chatting, J. Comput. Syst. Sci. 79 (5) (2013) 556–572.
[52] Z. Yan, M. Wang, P. Zhang, A scheme to secure instant community data access based on trust and contexts, in: IEEE International Conference on Computer and Information Technology, IEEE, Xi'an, China, 2014, pp. 646–651.
[53] Z. Yan, P. Zhang, A. Vasilakos, A survey on trust management for internet of things, J. Netw. Comput. Appl. 42 (2014) 120–134.
[54] J. Yao, A. Vasilakos, W. Pedrycz, Granular computing: Perspectives and challenges, IEEE Trans. Cybern. 43 (6) (2013) 1977–1989.
[55] S. Yoon, K. Kim, J. Hong, S. Kim, S. Park, A community-based sampling method using DPL for online social networks, Inf. Sci. 306 (2015) 53–69.
[56] K. Yu, Y. Gao, P. Zhang, M. Qiu, Design and architecture of dell acceleration appliances for database (DAAD): A practical approach with high availability guaranteed, in: IEEE 17th International Conference on High Performance Computing and Communications, IEEE, 2015, pp. 430–435.
[57] Y. Yu, J. Ni, M. Au, Y. Mu, B. Wang, H. Li, Comments on a public auditing mechanism for shared cloud data service, IEEE Trans. Serv. Comput. 8 (6) (2015) 998–999.

Please cite this article as: Y. Li et al., Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences (2016), http://dx.doi.org/10.1016/j.ins.2016.09.005