International Journal of Engineering

Journal Homepage: www.ije.ir

An Efficient Secret Sharing-based Storage System for Cloud-based Internet of Things

H. Bypour[a], M. Farhadi*[a], R. Mortazavi[b]

[a] *School of Mathematics and Computer Science, Damghan University, Damghan, Iran*
[b] *School of Engineering, Damghan University, Damghan, Iran*

PAPER INFO

Paper history:
Received 23December 2018
Received in revised form 15 May 2019
Accepted 05 July 2019

Keywords:
Internet of Things
Cloud Storage
Secret Sharing Scheme
Aggregate Data

ABSTRACT

Internet of things (IoTs) is the newfound information architecture based on the internet that develops
inte ns between objects and services in a secure and reliable environment. As the availability of
art devices rises, secure and scalable mass storage systems for aggregate data is required in IoTs
ons. In this paper, we propose a new method for storing aggregate data in IoTs by the use of
shold secret sharing scheme in the cloud storage. In this method, original data is divided into t
at each block is considered as a share. The edge server does not send these shares (blocks)
through the secure channel) to cloud service providers (CSPs). Rather, the edge server hides
s (blocks) with XORing two secret values and publishes the result. Indeed, with this method,
SPs has an amount of block information.This scheme is also verifiable, i.e., in the verification
ch CSP can verify its quasi-share. Moreover, before data retrieval, the edge server checks the
ss of provided quasi-share from CSPs of an authorized group. Also, the proposed scheme is
since new data can be inserted or part of the original data can be deleted, without changing
is worth noting that the proposed scheme is more efficient compared with the other scheme
vy and complex computation is not required.

***doi**: 10.5829/ije.2019.32.08b.07*

1. INTRODUCTION

The concept of the internet of things was first introduced by Ashton [1] in 1999. He describes a world in which everything has its own digital identity and allows computers to manage them. The most important feature of IoTs is the ability to connect various types of objects to the virtual world.

The model of IoTs and its standards have been reviewed and surveyed in several literature studies [2-8]. Miraz et al. [5] discussed the the Internet of Things (IoTs), Internet of Everything (IoE), and Internet of Nano Things (IoNTs). They have distinguished the difference between IoTs and IoE which are wrongly considered to be the same by many people. Moreover, Lin et al. [4], first explore the relationship between Cyber-Physical Systems (CPS) and IoTs. Then they present existing architectures, enabling technologies, and security and privacy issues in IoTs to enhance the understanding of the state of the art IoTs development.

The challenges in IoTs have been addressed in several studies. Andra et al. [9] presented security vulnerabilities and challenges in IoTs and explored the security requirements for IoTs and provided a classification of the security challenges in IoTs systems using a new unique classification method consisting of four classes of attacks: physical, network, software, and encryption attacks. Moreover, Botta et al. [10] presented the integration of cloud computing and IoTs. In addition, Samuel [6] reviewed connectivity challenges in the IoTs-smart home. Also Wei et al. [11] presented survey work on the challenges issues and opportunities in IoTs. Generally, the IoTs system requires confidentiality, integrity, authentication and access control. Privacy and access control are also the major challenges of IoTs [12].

As the availability of many smart devices rises, fast and easy access to data as well as sharing more information is felt. Moreover, secure and scalable mass storage systems for aggregate data are required in IoTs applications [4]. Mollah et al. [13] proposed a new

*Corresponding Author Email: *farhadi@du.ac.ir* (M. Farhadi)

cryptographic scheme that smart devices can share data securely with others at the edge of cloud-assisted IoTs with checking the integrity of decrypted data in data sharing and downloading phase. Furthermore, they proposed data-searching scheme to search desired data/shared data by authorized users on storage where all data are in encrypted form.

One of the methods used in the IoTs is to employ the secret sharing scheme [14]. A secret sharing scheme is designed to safeguard a secret by splitting it into shares and distributing them among a group of participants. In 1979, (t, n)-threshold secret sharing schemes were proposed by Blackley [15] and Shamir [16], independently. In a (t, n)-threshold secret sharing scheme, a secret can be shared among n participants such that t or more participants can reconstruct the secret, but $t-1$ or fewer participants can not share it.

To adapt the IoTs applications, a cloud service provider offers rapid access to flexible, low-cost resources. Cloud computing is the most cost-effective method for utilization, protection, and upgrading the program and data. The advantage of using cloud computing is the almost unlimited storage space. Therefore, there is no need to concern about possible space shortages and to increase storage space. Also, since the information is stored in the cloud, preparing a backup version and restoring the information is much easier than storing the same information on a physical device. Therefore, most cloud service providers compete for data retrieval.

Asadi and Hamidi [17] point to the privacy issues of big data distributed in the cloud computing and analyze the privacy issue with the Petri model. Chen et al. [18] and Shen et al. [19] used the revised Blakley's and Shamir's secret sharing schemes, respectively, in a secure distributed file system. Then, Jiang et al. [14] proposed a secure and scalable storage system for aggregate data in IoTs, using the method proposed by Shen et al. [19]. Jiang et al. [14] introduced big data with M bytes which was divided into blocks with $T-1$ bytes for storage. Then, for each block, a polynomial of degree $T-1$ is generated, and each byte of each block is considered as a coefficient of a polynomial. Moreover, in this scheme, new data can be inserted, or part of the original data can be deleted, without changing shares.

In this paper, we proposed a new scheme based on the method Jiang et al. [14] for storing aggregate data in IoTs in cloud storage. In this scheme, we use a (t, n)-threshold secret sharing scheme. Our proposed scheme has the following properties:

1. The edge server divides the data into t blocks, and each block is considered as a share. However, the edge server does not send these shares (blocks) directly (through the secure channel) to CSPs, but the shares are generated by published information.
2. The correctness of published information about CSPs can be verified by CSPs in the verification phase.
3. In the data retrieval phase, before the data retrieval, the correctness of provided information by t CSPs checked by the edge server. Thus, the edge server can prevent the cheating of some malicious CSPs.
4. The edge server is responsible for receiving, sharing and retrieving data. However, if the security of the edge server is compromised, then t CSPs can retrieve the data in collaboration with each other and with the information, they had.
5. The new data can be inserted or part of original data can be deleted, without changing shares.
6. We have only simple and easy calculations of the hash function, and we only use "$\oplus$" (bitwise exclusive OR) and "$||$" (concatenation) operators in the calculations. These operators make our scheme less costly and more efficient than other schemes [13, 14].

The rest of the paper is structured in the following sections: in section 2 we provide some definitions. In section 3, we propose the new method for storing aggregate data of IoTs applications in cloud storage. Section 4 involves analyzing the proposed method. In this section, we will describe some of the features and performance of the proposed scheme. Finally, section 5 concludes our paper.

2. PRELIMINARIES

2. 1. Overall System Architecture

Data owner The data owner is the possessor of the sensitive data that he/she wants to store his/her data in the cloud storage environment. He/she registers in the cloud account and uploads his/her data by PC or laptop or smart devices. Then, whenever necessary, he/she will be able to access the data by requesting the data in his/her cloud account.

Edge server This part of the cloud receives data from the data owner and, for storing it, divides the data into shares by the secret sharing scheme. We do not have absolute trust in this party, which means that the edge server may be misled in generating and publishing values.

Cloud Service Provider (CSP) This party receives a share from the edge server. The CSPs of an authorized subset send their shares to the edge server for retrieving the data. We do not have trust in this party, which means that this party maybe sends an invalid share to the edge server in the data retrieval phase.

Bulletin Board (BB) One of CSP is considered as a bulletin board, which public values published in it. We suppose that only the edge server can publish the values on the bulletin board, and only it can change the values on the bulletin board, and the other (CSPs) can only view the published values.

2. 2. Security Properties

Correctness If edge server and CSPs act honestly, the data retrieve by shares of any authorized subset of CSPs.

Verifiability Each CSP must be able to check the accuracy of its share. Furthermore, before retrieving data, the edge server must be able to verify the accuracy of the shares received by the CSPs in order to prevent the cheating of CSPs.

Secrecy The basic requirement is that an adversary cannot learn any information about the data, or it is impossible for any collusion of less than t CSPs to obtain any information about the data.

2. 3. Cryptographic Method

We say that hash function H is cryptographically secure if H satisfies the following conditions:

- The hash function should be preimage resistant, i.e., for a given output y of H, it should be difficult to find a message m such that $y = H(m)$.
- The hash function should be second preimage resistant, i.e., for a given m_1, it should be difficult to find a message $m_2 \neq m_1$ such that $H(m_2) = H(m_1)$.
- The hash function should be collision resistant, i.e., it should be difficult to find two different messages m_1 and m_2 such that $H(m_1) = H(m_2)$.

The difficulty of finding a collision depends on the size of the hash value.

2. 4. Threat Model

Insider threats Malicious insiders such as CSPs that want to access/disclose/modify the stored data.

Outsider threats Outside intruders are those who want to access the data, alone or with the collaboration of some unauthorized subsets of CSPs.

3. THE PROPOSED SCHEME

In this section, we describe our proposed scheme. We need some notations and values in Table 1.

In the proposed scheme, we assume that each CSP_i has ID_{1i}, ID_{2i} which receives them with $\{ID'_{2i}\}_{CA} = H(ID_{2i})$ from CA (certificate authority). Also, CA sends ID_{1i} to the edge server.

The steps of the proposed scheme are as follows:

- **Submitting the data**
  - **Registration**

    As shown in Figure 1, the data owner selects a username and registers her/his information. Then, he/she receives a password (note that the data owner can change the password).

TABLE 1. The notations used in the proposed scheme

Notation	Meaning
D	Data
N	The size of data D
n	The number of cloud service providers (CSPs)
t	The threshold of CSPs
m	The size of t blocks of data D
$H(\cdot)$	The secure hash function
$\{0,1\}^*$	The set of all binary strings of arbitrary finite bit length
$\{0,1\}^q$	The set of all binary strings of fixed length q
$\bar{D}$	The hash value of the data D
CSP_i	The ith cloud service provider
B_i	The ith block with size q
w	The number of authorized groups
$\|\|$	Concatenation
$\oplus$	Bitwise exclusive OR operator
$\bar{B}_{ij}$	The hash value of (Block $i\|\|j$)
$\bar{D}$	The hash value of data D
d_i	The quasi share of CSP_i
$\hat{d}_j$	The public value
v_{ij}	The public value (value of masked $\bar{B}_{ij}$)
H_j	The hash value of $d_1\|\|d_2\|\| \dots \|\|d_t\|\|j$
s_{ij}	The public value (value of masked block B_i)
T	The chosen randomly value by the edge server
I_i	The hash value of $T\|\|ID_{1i}\|\|ID_{2i}$
$\tilde{d}_i$	The value of masked quasi-share d_i
d'_i	The hash value of quasi-share d_i

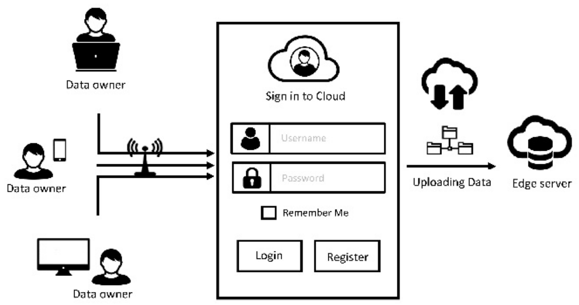

Figure 1. Registration and submission of data by the data owner

- **Login and submitting**

1. The data owner logs into a nearby edge server from a smart device using the username and password and submits the data S.
2. The data owner also submits some related keywords of the data such that any authorized recipient users are allowed to view to find the data. In this case, the

data owner can introduce some users to access data. There are two cases when users of the group want to access the data using keywords:

- Users of the group are authorized to access the data. In this case, authorized users request the data by providing keywords. Then, the edge server provides the data to them.
- The data owner revokes access of some users to the data in the group. In this case, the data owner revokes access of some users (U_i) by providing (ID_i, keywords, Neg) to the edge server (note that we assume Neg is a sign of revocation of access to the data). Once, when the user U_i requests the data by providing the keyword, the edge server checks the user U_i access to the data. In this case, the edge server will display "unauthorized access" to the user U_i.

• **Construction**

1. After submitting the data D by the data owner, the edge server first publishes an access structure $\Gamma = \{A_1, A_2, \dots, A_w\}$ on the BB, where A_j $(j = 1, 2, \dots, w)$ is an authorized group. Then, the edge server splits the data into t blocks with size m. If $t \nmid |\text{Data}|$, then the edge server appends the randomly generated padding strings to the end of the data such that $t \mid |\text{Data} \,||\, \text{padding}|$.
2. Suppose $|D| = |\text{Data}| = N$ or $|D| = |\text{Data} \,||\, \text{padding}| = N$ and $t \mid N$ .So

$$D = \overbrace{a_{11}a_{12}\dots a_{1m}||a_{21}a_{22}\dots a_{2m}||\dots||a_{t1}a_{t2}\dots a_{tm}}^{N}$$

$Block\ 1$: $a_{11}a_{12}\dots a_{1m}$

$Block\ 2$: $a_{21}a_{22}\dots a_{2m}$

$\vdots$

$Block\ t$: $a_{t1}a_{t2}\dots a_{tm}$

3. The edge server chooses a secure hash function $H: \{0,1\}^* \longrightarrow \{0,1\}^q$ and computes

$$\bar{D} = H(D) \tag{1}$$

4. Now, the edge server acts based on the following cases:

- **If |Block *i*| < *q***

1. The edge server appends the randomly generated padding strings to the end of each block such that the length of the block equals to q.

$$B_i = \text{Block } i \,||\, \text{padding } i, \quad i = 1, 2, \dots, t \tag{2}$$

$$\Rightarrow |B_i| = q$$

$$\bar{B}_{ij} = H(\text{Block } i||j) \tag{3}$$

2. Each CSP_i sends $\mathcal{ID}_i = ID_{1i} \oplus ID_{2i}$, $\{ID'_{2i}\}_{CA}$ to the edge server. The edge server first obtains $ID_{2i} = \mathcal{ID}_i \oplus ID_{1i}, ID''_{2i} = H(ID_{2i})$ and then compares ID''_{2i} with $\{ID'_{2i}\}_{CA}$. If two hash values are equal, then the edge server accepts ID_{2i}, otherwise rejects it. After this, the edge server obtains for every CSP_i in an authorized group A_j

$$\forall d_1, d_2, \dots, d_n \in \{0,1\}^q, \ \exists\ \hat{d}_j \in \{0,1\}^q \text{ s.t.}$$
$$\bar{D} = \oplus_{csp_i \in A_j} d_i \oplus \hat{d}_j \tag{4}$$

$$v_{ij} = \bar{B}_{ij} \oplus \bar{D} \tag{5}$$

$$H_j = H(d_1||d_2||\dots||d_t||j) \tag{6}$$

$$s_{ij} = B_i \oplus \bar{B}_{ij} \oplus H_j \tag{7}$$

$$I_i = H(T||ID_{1i}||ID_{2i}),\ T \in_R \{0,1\}^q \tag{8}$$

$$\tilde{d}_i = d_i \oplus I_i \tag{9}$$

$$d'_i = H(d_i) \tag{10}$$

where $i = 1, 2, \dots, t, j = 1, 2, \dots, w$.

3. The edge server sends $\tilde{d}_i$ to CSP_i and publishes $T, \hat{d}_j, d'_i, v_{ij}, s_{ij}$, and Hash function H on the BB ($T, \hat{d}_j, d'_i$ for CSP_i and v_{ij}, s_{ij} for itself). Indeed, each CSP_i just gets d_i as quasi-share and in the Verification phase, it just needs to verify the validity of d_i. Then, the edge server stores the size of paddings.

Figure 2 shows the construction phase of the proposed scheme.

- **If |Block *i*|= *q* or |Block *i*|= *kq***

1. In case |Block i|= q, it is not necessary to perform the padding for Block i in Eq. (2).
2. In case |Block i|= kq, the edge server considers

$$k\bar{B}_{ij} = \overbrace{\bar{B}_{ij}||\bar{B}_{ij}||\dots||\bar{B}_{ij}}^{k}. \tag{11}$$

Similar to Equation (11), the edge server considers $k\bar{D}, kH_{ij}$ instead of $\bar{D}, H_{ij}$. Then, the edge server performs the rest of the calculation in the same way as the case |Block i|< q.

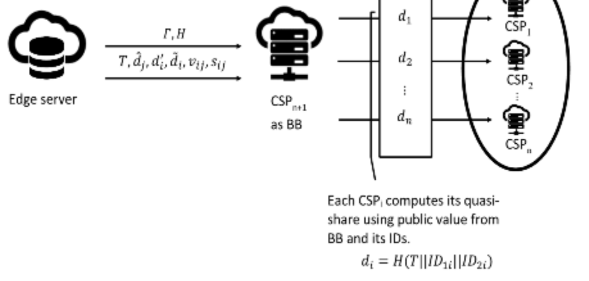
Edge server
Γ, H
$T, \hat{d}_j, d'_i, \tilde{d}_i, v_{ij}, s_{ij}$
CSP$_{n+1}$ as BB
d_1
d_2
d_n
CSP$_1$
CSP$_2$
CSP$_n$
Each CSP$_i$ computes its quasi-share using public value from BB and its IDs.
$d_i = H(T||ID_{1i}||ID_{2i})$

Figure 2. The Construction phase of the proposed scheme

- **If |Block i|> q**
  Suppose

$(k-1)q <$|Block i|$< kq$.

Note that the size of the block here is greater than q and is not divisible by q. In this case, the edge server performs as follows:

$$\forall i = 1,2,\dots,t \;\; B_i = \text{Block } i||\text{padding } i \quad \text{s.t. } |B_i| = kq \tag{12}$$

Then, the edge server performs the rest of the calculation in the same way as case |Block i|= kq.

- **Verification**
  Each CSP_i computes

$I_i = H(T||ID_{1i}||ID_{2i})$,

$d_i = \tilde{d}_i \oplus I_i$

$d_i'' = H(d_i)$

and then checks

$$d_i'' \overset{?}{=} d_i'. \tag{13}$$

If Equality (13) holds, the published hash value d_i' is valid. Otherwise d_i' is invalid.

After successful verification, the edge server deletes $ID_{2i}, i = 1,2,\dots,n$.

- **Data requesting and retrieval**
  We only show data retrieval for the first case where |Block i|$< q$, and the two remaining cases are similarly obtained.
  1. If data owner requests the data (or an authorized user requests the data by searching keywords), then CSPs of an authorized group j provide their quasi-shares to edge server.
  2. After sending t quasi-shares d_i by CSP_i $(i = 1,2,\dots,t)$, the edge server obtains and checks

$$d_i'' = H(d_i) \tag{14}$$

$$d_i'' \overset{?}{=} d_i'. \tag{15}$$

The edge server accepts quasi-share d_i, if Equality (15) holds, and rejects it otherwise.

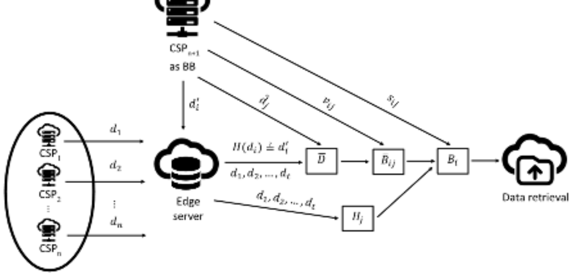

Figure 3. Data retrieval phase of proposed scheme

3. After accepting t quasi-shares, the edge server computes

$$\bar{D} = \oplus_{csp_i \in A_j} d_i \oplus \hat{d}_j \tag{16}$$

$$\bar{B}_{ij} = v_{ij} \oplus \bar{D} \tag{17}$$

$$H_j = H(d_1||d_2||\dots||d_t||j) \tag{18}$$

$$B_i = s_{ij} \oplus \bar{B}_{ij} \oplus H_j. \tag{19}$$

Then, the edge server appends the obtained blocks together for retrieval.

Figure 3 shows the Data retrieval phase of the proposed scheme.

- **Inserting new data**
  Suppose that there is new data to be added to the original data. So, there are four cases:

1. **Inserting new data before Block 1**

The new discrete block(s) is/are generated. There are three cases:

- If

|new data| < |Block 1|,

then the edge server randomly appends a new padding to the new data block such that

|new data || padding|= m.

- If

|new data|= |Block 1|,

then the edge server considers the new data as the new block of size m.

- If

|new data| > |Block 1|,

then the edge server splits the new data into blocks of size m. If

|last generated new block| < |Block 1|,

then the edge server randomly appends a new padding in the same way as the first instance.
In general, in this case, there are more than t blocks. Therefore, at least $t+1$ CSPs can retrieve data.

2. **Inserting new data between Block i data**

The edge server splits the Block i with inserted data into new blocks of size m. If

|last generated new block| < |Block i|,

then the edge server randomly appends a new padding in the same way as the first instance of the previous case.

3. **Inserting new data between Block i and Block $i+1$**

The new discrete block(s) is/are generated similar to case 1.

4. **Inserting new data after the last Block**
   The new discrete block(s) is/are generated similar to case 1.

- **Deleting part of the data**
  Suppose that part of data is deleted. So, there will be three cases.
  1. **Deleted data is just part of Block i; |deleted data| < |Block i|**
  In this case, the edge server performs the padding again only for the block that part of its data has been deleted.
  2. **Deleted data is part of Block i and Block $i+1$; |deleted data| < |Block i, Block $i+1$|**
  In this case, the edge server performs the padding again only for Block i and Block $i+1$.
  3. **|deleted data| > |Block i|**
  In this case, less than t blocks remain. So, the deleted block(s) should be generated. If the number of deleted Blocks < |Block i|, then the edge server selects an adjacent block of deleted data and splits it into blocks of size 1. Note that

  $1 \leq$ |last generated new Block $< m$.

  Then, edge server appends padding to the newly generated blocks. Now, if

  the number of deleted Blocks > |Block i|,

  then the edge server selects several adjacent blocks of deleted data for splitting and generates new blocks instead of deleted blocks similar before.

4. ANALYSIS OF OUR PROPOSAL

4. 1. Security Analysis In this section, we will analyze the features of our proposal.

Theorem 4.1. *Any collusion of less than t CSPs cannot obtain any information about the data.*

Proof. Without loss of generality, suppose $CSP_1, CSP_2, \ldots, CSP_{t-1}$ (malicious insiders) with quasi-shares $d_1, d_2, \ldots, d_{t-1}$, respectively, intend to retrieve the data without CSP_t. To do this, malicious insiders should obtain blocks $B_1, B_2, \ldots, B_t$. However, to obtain these blocks, they need values $\bar{B}_{ij}$, H_j, which values $\bar{B}_{ij}$, H_j are derived from Equations (5) and (6), respectively. That is, they must first obtain the quasi-share d_t. Malicious insiders can try to derive quasi-share of CSP_t by published value d_i' in the construction phase. Since the hash function H is one-way, they cannot obtain the quasi-share d_t from public information d_i'.

Theorem 4.2. *The edge server cannot cheat by publishing invalid values in the Construction phase. Indeed, its cheating is detectable.*

Proof According to the verification phase, if the edge server publishes invalid d_i', then Equality (13) does not hold. Indeed, since the hash function H is second preimage resistant, the edge server cannot find the second preimage $\bar{d}_i$ such that $d_i' = H(\bar{d}_i) = H(d_i)$. Moreover, since values v_{ij}, s_{ij} are required for data retrieval, so the edge server must publish valid values of v_{ij}, s_{ij} in the construction phase. Otherwise, in the data retrieval phase, the original data will not be restored, and this will be the edge server error. ■

Theorem 4.3. *If some malicious $CSPs$ provide fake shares to prevent retrieval of the main data, then their cheating is detectable.*

Proof According to the data retrieval phase, if CSP_i provides invalid quasi-share d_i, similar to proof of Theorem 4.2, Equality (15) does not hold. ■

Theorem 4.4. *Under the secure hash function, outside intruders cannot get any information about the data.*

Proof Suppose outside intruders want to achieve blocks of data by public information $T, \hat{d}_j, d_i', v_{ij}, s_{ij}$. But, according to proof of Theorem 4.1, outside intruders need to get a quasi-shares of t CSPs to get blocks and retrieve the original data. But since H is a one-way function, they can not get t quasi-shares and retrieve data using public values d_i'. ■

Remark Our proposal is scalable because we can add new data or delete part of the original data without changing all the shares. When inserting new data, previous shares don't change and only some new shares are generated. Also, when part of the original data is deleted, only some of the blocks are affected by the deletion of the data that are subject to change.

4. 2. Performance Analysis In this section, in Tables 2 and 3, we consider two schemes for comparison with our scheme. One of the reported schemes [13] is based on the cryptographic mechanisms and the other [14] is based on the secret sharing scheme.

The scheme introduced by Mollah [13] used RSA and AES cryptosystems. Since the computational complexity of the RSA encryption is $O(n^3)$, the computational complexity of the scheme becomes $O(n^3)$ (It is notable that the symmetric cryptosystem AES has a much lower computational complexity, so we ignore it).

The scheme introduced by Jiang et al. [14] splits the data into several blocks such that for each block, a polynomial of degree $t-1$ is formed. If we assume that there are l blocks, then we have l polynomials. To retrieve data, each t shares together retrieves these l blocks and the scheme requires $O(lt^2)$ computations.

The proposed scheme is established based on the XOR operator, concatenation, and hash functions. The construction phase requires $O(t)$ computations since it consists of $(5w+9)t+3$ XOR operators, concatenations and hash function calls. The verification phase comprises of (n) XOR operators, $(2n)$ concatenations and calls the hash function (n) times. Therefore, it is accomplished in $O(n)$. Similarly, the data retrieval phase comprises of $(2t)$ XOR operators, (t) concatenations and calls the hash function (t) times. Therefore it is accomplished in $O(t)$.

4. 3. Experimental Evaluations

We implemented and compared the runtime of the proposed method and the technique introduced by Jiang et al. [14]. In both experiments, the same 10000 secret messages each of length 70 bytes are sent to participants (CSPs) and reconstructed using t shares. Each experiment is repeated 10 times and the average required time per message is measured. Figure 4 shows the results of the methods for $t = 3, 4, 5, 6, 7, 8, 9$. The methods are implemented in C#.Net on a regular PC with an Intel G3220 3.0 GHz CPU and Windows 10 operating system. The results confirm the effectiveness of the proposed method in all cases. This performance gain is the result of using hash functions and simple operators such as XOR and concatenation instead of heavy matrix or polynomial computations in finite fields. It is notable that the proposed method also includes the verification step that is not part of the technique defined by Jiang et al. [14].

TABLE 2. Comparisons between the proposed scheme with other related schemes

Feature	Ref. [13]	Ref. [14]	Our scheme
Data privacy	Yes	Yes	Yes
Data availability	Yes	Yes	Yes
Verifiability	No	No	Yes
Data scalability	No	Yes	Yes
Resist cheating by dishonest CSPs	Yes	No	Yes
Type of cloud	Single cloud	Multi-cloud	Multi-cloud
Method used	AES and RSA	Polynomial	Hash-based Secret sharing
Need secure channel	No	Yes	No
Have public value	No	No	Yes

TABLE 3. Comparison of computational complexity

Feature	Ref. [13]	Ref. [14]	Our scheme
Construction	$O(n^3)$	$O(ln)$	$O(t)$
Verification	___	___	$O(n)$
Retrieval	$O(n^3)$	$O(lt^2)$	$O(t)$

5. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new verifiable and scalable cloud storage for aggregate data in IoTs. In this scheme, the edge server divided the data into t blocks and each block was considered as a share. But these blocks were hidden for each CSP. Indeed, the edge server obtained a quasi-share for each CSP and published the hash value of it. Then in the data retrieval phase, the edge server retrieved the blocks of data by quasi-share of CSPs of an authorized group. In the verification phase, each CSP obtained the hash value of its quasi-share and compared it with published hash value by the edge server to observe whether the valid hash value has been published to it by the edge server. Moreover, before data retrieval, the edge server checked the correctness of provided quasi-shares by t CSPs. Therefore, the fault of CSPs in providing invalid quasi-shares was detectable. Since blocks of data are hidden using the quasi-shares, the confidentiality of data is maintained. Thus, this scheme is safe against attacks of malicious insiders and outsiders. Also, in our scheme, new data can be inserted or part of the original data can be deleted, without changing all shares. Furthermore, we showed that our scheme is more efficient than some other schemes because, in this scheme, we had only simple and easy calculations of the hash function, and we only used "⊕" (bitwise exclusive OR) and "||" (concatenation) operators in the calculations.

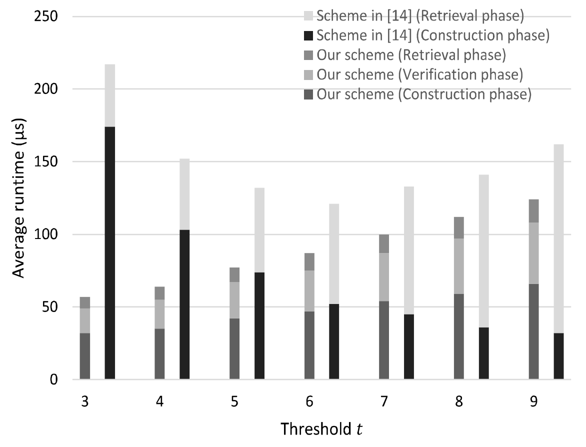

Figure 4. The execution time of our scheme compared to the scheme introduced by Jiang et al. [14]

In this paper, we proposed a scheme in which the edge server is trusted. But in the real world, the server may not be absolutely reliable. In this way, it can exploit user information and documents. So, we are looking to propose a scheme in which the edge server and CSPs are not reliable, and the user's file and information in cloud environments are stored in such a way that the servers and CSPs do not access its content in any way.

6. REFERENCES

1. Ashton, K., "That 'internet of things' thing", ***RFID Journal***, Vol. 22, No. 7, (2009), 97-114.
2. Belkeziz, R. and Jarir, Z., "A survey on internet of things coordination", in International Conference on Systems of Collaboration., (2016), 1-6.
3. Darshan, K. and Anandakumar, K., "A comprehensive review on usage of internet of things (iot) in healthcare system", in 2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), IEEE., (2015), 132-136.
4. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H. and Zhao, W., "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications", ***IEEE Internet of Things Journal***, Vol. 4, No. 5, (2017), 1125-1142.
5. Miraz, M.H., Ali, M., Excell, P.S. and Picking, R., "A review on internet of things (iot), internet of everything (ioe) and internet of nano things (iont)", in 2015 Internet Technologies and Applications (ITA), IEEE., (2015), 219-224.
6. Samuel, S.S.I., "A review of connectivity challenges in iot-smart home", in 2016 3rd MEC International conference on big data and smart city (ICBDSC), IEEE. Vol., No. Issue, (2016), 1-4.
7. Suresh, P., Daniel, J.V., Parthasarathy, V. and Aswathy, R., "A state of the art review on the internet of things (iot) history, technology and fields of deployment", in 2014 International conference on science engineering and management research (ICSEMR), IEEE., (2014), 1-8.
8. Tayeb, S., Latifi, S. and Kim, Y., "A survey on iot communication and computation frameworks: An industrial perspective", in 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC), IEEE., (2017), 1-6.
9. Andrea, I., Chrysostomou, C. and Hadjichristofi, G., "Internet of things: Security vulnerabilities and challenges", in 2015 IEEE Symposium on Computers and Communication (ISCC), IEEE., (2015), 180-187.
10. Botta, A., De Donato, W., Persico, V. and Pescapé, A., "On the integration of cloud computing and internet of things", in 2014 International Conference on Future Internet of Things and Cloud, IEEE., (2014), 23-30.
11. Wei, W., Yang, A.T. and Shi, W., "Security in internet of things: Opportunities and challenges", in 2016 International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI), IEEE., (2016), 512-518.
12. Roman, R., Zhou, J. and Lopez, J., "On the features and challenges of security and privacy in distributed internet of things", ***Computer Networks***, Vol. 57, No. 10, (2013), 2266-2279.
13. Mollah, M.B., Azad, M.A.K. and Vasilakos, A., "Secure data sharing and searching at the edge of cloud-assisted internet of things", ***IEEE Cloud Computing***, Vol. 4, No. 1, (2017), 34-42.
14. Jiang, H., Shen, F., Chen, S., Li, K.-C. and Jeong, Y.-S., "A secure and scalable storage system for aggregate data in iot", ***Future Generation Computer Systems***, Vol. 49, (2015), 133-141.
15. Blakley, G.R., "Safeguarding cryptographic keys", in Proceedings of the national computer conference. Vol. 48, (1979), 313-317.
16. Shamir, A., "How to share a secret", ***Communications of the ACM***, Vol. 22, No. 11, (1979), 612-613.
17. Asadi, F. and Hamidi, H., "An architecture for security and protection of big data", ***International Journal of Engineering***, Vol. 30, No. 10, (2017), 1479-1486.
18. Chen, S., Chen, Y., Jiang, H., Yang, L.T. and Li, K.-C., "A secure distributed file system based on revised blakley's secret sharing scheme", in 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, IEEE., (2012), 310-317.
19. Shen, F., Jiang, H. and Xu, Z., "On post-generation data operations in secure distributed storage systems with internal padding", in 2010 10th IEEE International Conference on Computer and Information Technology, IEEE., (2010), 2698-2705.

An Efficient Secret Sharing-based Storage System for Cloud-based Internet of Things

H. Bypour[a], M. Farhadi[a], R. Mortazavi[b]

[a] *School of Mathematics and Computer Science, Damghan University, Damghan, Iran*
[b] *School of Engineering, Damghan University, Damghan, Iran*

PAPER INFO

Paper history:
Received 23December 2018
Received in revised form 15 May 2019
Accepted 05 July 2019

Keywords:
Internet of Things
Cloud Storage
Secret Sharing Scheme
Aggregate Data

چکیده

اینترنت اشیاء، معماری اطلاعات نوظهور مبتنی بر اینترنت است که تعاملات بین اشیاء و سرویس‌ها (خدمات) را در یک محیط امن و قابل اطمینان توسعه می‌دهد. چون دسترسی زیاد دستگاه‌های هوشمند روبه افزایش است، سیستم‌های ذخیره‌سازی انبوه امن و مقیاس‌پذیر برای داده‌های جمع شده در برنامه‌های IoTs مورد نیاز است. در این مقاله، یک روش جدید برای ذخیره‌سازی داده‌های جمع‌شده در IoTs با استفاده از تسهیم راز آستانه‌ای (t,n) در ذخیره‌سازی ابر پیشنهاد می‌کنیم. در این روش، داده اصلی به t بلوک تقسیم می‌شود طوری‌که هر بلوک به‌عنوان یک سهم در نظر گرفته می‌شود. سرور لبه این سهم‌ها (بلوک‌ها) را به‌طور مستقیم (از طریق کانال خصوصی) برای سرویس‌های ارائه‌دهنده خدمات (CSPها) ارسال نمی‌کند. بلکه سرور لبه این سهم‌ها (بلوک‌ها) را با XOR کردن با دو مقدار مخفی، پنهان می‌کند و مقدار بدست‌آمده را منتشر می‌کند. در واقع با این روش، هیچ یک از CSPها اطلاعاتی درباره بلوک‌ها ندارد. این طرح همچنین تأییدپذیر است. یعنی، در مرحله تأییدپذیری، هر CSP می‌تواند درستی شبه‌سهم خود را تأیید کند. بعلاوه، قبل از بازیابی داده، سرور لبه درستی شبه‌سهم‌های ارائه شده توسط CSPها را بررسی می‌کند. همچنین طرح پیشنهادی مقیاس‌پذیر است، چون، بدون تغییر دادن سهام، می‌توان داده جدید اضافه و یا بخشی از داده اصلی را حذف کرد. قابل توجه است که طرح پیشنهادی در مقایسه با طرح‌های دیگر کارآمدتر است، زیرا محاسبات سنگین و پیچیده مورد نیاز نیست.

doi: *10.5829/ije.2019.32.08b.07*