Computers and Electrical Engineering 000 (2016) 1–14

Contents lists available at ScienceDirect

Computers and Electrical Engineering

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

Cellular automata based secure distributed storage scheme with integrity proof ☆

Yousheng Zhou [a,b], Feng Wang [c], Fei Tang [a,*], Xiaojun Wang [d]

[a] College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[b] School of Computer Science, Chongqing University, Chongqing 400044, China
[c] College of mathematical sciences, De Zhou University, Shandong 253023, China
[d] School of Electronic Engineering, Dublin City University, Dublin, Ireland

ARTICLE INFO

Article history:
Received 9 September 2015
Revised 16 September 2016
Accepted 2 November 2016
Available online xxx

Keywords:
Cellular automata
Threshold storage
Integrity proof
Cloud security

ABSTRACT

The cloud storage service, widely used in daily life due to its convenience, can sometimes suffer from availability and confidentiality problems. For instance, data in cloud storage can be damaged by hardware failures or malicious destruction, or even be exposed to unauthorized parties, and this poses a big risk for user data stored in cloud. To overcome these problems and challenges, a Cellular Automata based secure Distributed storage scheme with Integrity Proof, termed CAD-IP, is proposed in this paper. CAD-IP leverages threshold based storage service to provide robustness and confidentiality of user private data. Homomorphic hashing is integrated into the proposed scheme, which facilitates the verifier to check the integrity of the data on servers. A sampling strategy that greatly reduces the computation and communication cost is adapted in the proposed scheme. Analysis demonstrates that the proposed scheme can not only achieve perfectness, confidentiality and unforgeability, but also enable the verifier to effectively detect any modification or deletion of their file shares. Meanwhile, the proposed CAD-IP scheme supports dynamic update over the stored file shares without downloading and re-uploading the entire file shares.

1. Introduction

Cloud computing is seen as the next wave of information technology for individuals and organizations, which treats computing as a service rather than a product, enabling users to access and share a wide variety of applications, data, and resources through an universal interface [1]. This new economic computing model is commonly referred to as cloud computing and includes various types of services such as: infrastructure as a service (IaaS), where a customer makes use of a service providers computing, storage or networking infrastructure; platform as a service (PaaS), where a customer leverages the providers resources to run custom applications; and finally software as a service (SaaS), where customers use software that is run on the providers' infrastructure [1]. Cloud computing has been widely applied into many fields, such as e-health wireless sensor networks [2], the management for mobile devices and so on [3]. Cloud computing has experienced exponen-

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. Z. Wang.

* Corresponding author.
E-mail address: tangfei@cqupt.edu.cn (F. Tang).

http://dx.doi.org/10.1016/j.compeleceng.2016.11.004
0045-7906/

tial growth over the last few years due to its convenience and pay-as-you-go charging model, and the growth is expected to increase over the next few years worldwide [4].

Cloud computing brings incredible convenience for both individuals and organizations to store their data in cloud, and users can access their data freely via some interface, for instance a browser, at any time anywhere, while they need not to care about the operating, monitoring and maintenance behind the data [5,6]. While the benefits of using a cloud infrastructure are clear, it introduces significant security and privacy risks. In most occasions, however, data are under the full control of the cloud service providers once users outsource their storage to the cloud. In reality, the services sometimes inevitably suffer from confidentiality, availability and robustness problems, and this renders customers cannot use them for reasons, such as service down, network connection banned, or malicious attack [7–9], which greatly reduces the availability of the cloud services and becomes one of the hurdles hindering wider adaptation of cloud computing. Although cloud storage has enormous promises, unless the issues of confidentiality and reliability are properly addressed many potential customers will be reluctant to make the move [10–12]. How to prevent private data in cloud accessed by unauthorized parties is a challenge for any cloud storage providers. Access control seems to be a suitable solution to this problem[13], however, data stored in cloud may be shared by multiple users and the cloud servers are not always trusted. Merely using access control is not effective, and cryptographic methods can be used to ensure confidentiality [14–16]. Wang et al. [17] proposed a hierarchical attribute based encryption scheme for cloud storage, Zhou et al. [18] proposed a role-based encryption (RBE) scheme that allows Role-Based-Access-Control (RBAC) policies to be enforced for the encrypted data stored in public clouds. Users who violate the rules of the shared data should be revoked by the data owner, re-encryption can be adopted to address it [19–22]. Do et al. [20] proposed a proxy re-encryption scheme to resolve it by dividing a data file into header and body. However, re-encryption commands can be intercepted when they are transferred over an untrusted network, Liu et al. [21,22] solved this problem by proposing a time-based re-encryption scheme, which enables the cloud servers to automatically re-encrypt data based on their internal clocks. Although the confidentiality of the data can be ensured using encryption, the user has to download the entire data from the server if the user wants to search the data. To address this problem, research has been conducted on keyword search over encrypted data [23,24,26]. With the construction of a special tree-based index structure, Xia et al. [23] presented a multi-keyword ranked search scheme over encrypted cloud data, which supports dynamic update operations. Fu et al. [24] proposed a searchable encryption scheme which supports both multi-keyword ranked search and parallel search based on Vector Space Model (VSM). In addition, Fu et al. [25,26] proposed some improved schemes based on the semantic relationship between concepts and the uni-gram to improve efficiency and accuracy in multi-keyword search.

The efforts stated above only focus on confidentiality and availability of stored data in cloud, however, robustness is not provided. Lin et al. [28] considered the problem of robustness and confidentiality with erasure code-based method, however, their scheme cannot provide integrity proof. Data loss can happen occasionally due to either natural reasons or malicious attacks. Though, backup technology is adopted by cloud service providers, the confidentiality cannot be ensured, since all the backups are the same and encrypted by the same private key, once the only secret key used for encryption is leaked or destroyed, the confidentiality of all the backups cannot be assured anymore. Provable data possession (PDP) [39] can efficiently address the problem of integrity, and it allows a user who has stored data to an untrusted server to verify whether the server possesses the original data without retrieving it. Many efforts on data integrity have been made in recent years, such as Zhu et al. [37] presented a cooperative PDP scheme based on homomorphic verifiable response and hash index hierarchy, which was claimed to have the properties of completeness, knowledge soundness, and zero-knowledge. However, Wang et al. [35] pointed out the scheme in [37] fails to provide knowledge soundness and it is vulnerable to cheating attacks. To facilitate the PDP in multicloud storage, Wang et al. [38] proposed an efficient identity-based distributed provable data scheme using the bilinear pairings. In order to deal with the issue of burdensome certificate management in Public Key Infrastructure(PKI) based cloud storage, Wang et al. [36] constructed a certificate-based remote data integrity checking model, which enables the cloud servers to detect the malicious clients. In addition, some other research focused on the extension of PDP [41–43].

Motivated by the characteristic distributed property of threshold secret sharing [29], a novel distributed storage scheme with integrity proof based on cellular automata [30] for cloud storage, i.e. the CAD-IP scheme, is constructed in this paper, which owns desirable security and performance features as follows,

(1) *Security*. Our CAD-IP scheme can simultaneously provide correctness, correctness, confidentiality, and reliability. Due to easy hardware implementation and its pseudorandom behavior, cellular automata have been widely used in cryptography [34,44–49]. Although there exist some research [34,44,45] on cellular automata based secret sharing, these schemes are not the desired solutions for distributed cloud storage since they cannot provide integrity proof. In our construction, threshold storage of the file ensures confidentiality and robustness, and it tolerates Byzantine failures [31], where a storage server may fail in arbitrary ways.
(2) *Performance*. By utilizing the homomorphic token [32], the presented scheme achieves dynamic operation on random sampled blocks, not all data blocks, to provide validation of integrity, which means only several bytes of hashing value would be transferred over the communication channel, and the client needs less storage to complete validation compared to previous threshold based storage schemes. Furthermore, no additional encryption/decryption or encoding/decoding operations when to provide confidentiality in our scheme. Above all, computation and communication cost are reduced significantly in our proposed CAD-IP.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

The rest of the paper is organized as follows: Section 2 introduces some preliminaries with respect to our scheme. Definitions of system model and security model for CAD-IP is given in Section 3. Then the detailed construction of CAD-IP is illustrated in Sections 4. Section 5 gives the security analysis and performance evaluations. Finally, the paper is concluded in Section 6.

2. Basic concepts

2.1. Global homomorphic hashing

Definition 1. A *homomorphism* is a map that preserves selected structure between two algebraic structures $\mathcal{X}, \mathcal{Y}$ exists a map $\varphi : \mathcal{X} \to \mathcal{Y}$, which satisfies

$$\varphi(x.y) = \varphi(x) \circ \varphi(y),$$

where . and $\circ$ are operations of $\mathcal{X}$ and $\mathcal{Y}$ respectively.

Definition 2. A *global homomorphic hashing* is a hash function which satisfies the requirements of homomorphism, the global homomorphic hashing h used in our scheme is from [15], which is defined as follows:

$$h : \chi \to \prod_{i=1}^{m'} g_i^{x_i} (\mathrm{mod} p), \tag{1}$$

where $\chi = (x_1, x_2, \ldots, x_{m'}), g_i = g^{r_i}, r_i \in Z_q$, g is the generator of the cyclic group G with the prime order q.

2.2. Cellular automata

One-dimensional cellular automata (CA for short) [30] is a discrete dynamic system which consists of an array of N identical objects called cells, and each cell has a state $S \in Z_q$, which can be updated synchronously in line with a local transition function f. The local transition function takes the previous states of a set of cells, including the cell itself, as input, and that cells set called its neighborhood. For convenience, denote the ith cell as $< i >$, and the symmetric neighborhood with radius r as $\mathcal{N}_i = (< i-r >, < i-r+1 >, \ldots, < i-1 >, < i >, < i+1 >, \ldots, < i+r-1 >, < i+r >)$, and the state of $< i >$ at time T as $a_i^{(T)}$. Then, the local transition function of the cellular automata with radius r can be denoted as follows,

$$a_i^{(T+1)} = f(a_{i-r}^{(T)}, \ldots, a_i^{(T)}, \ldots, a_{i+r}^{(T)}), \quad 1 \le i \le N-1, \tag{2}$$

or equally,

$$a_i^{(T+1)} = f(\mathcal{N}_i^{(T)}), \quad 1 \le i \le N-1, \tag{3}$$

where $\mathcal{N}_i^{(T)} \subset Z_q^{(2r+1)}$ denotes the states of $< i >$'s neighbor cells at time T. Note, if $i \equiv j (\mathrm{mod} N)$, then it means that $a_i^{(T)} = a_j^{(T)}$ to fit the well-defined dynamics of the CA. We call the vector $C^{(T)} = (a_0^{(T)}, a_1^{(T)}, \ldots, a_{N-1}^{(T)})$ be the configuration of CA at time T. Specially, $C^{(0)}$ is called *the initial configuration*. What is more, the sequence $\{C^{(T)}\}_{0 \le T \le k}$ is called*the evolution of* CA *with order k*. We denote the collection of all possible configurations of the CA as $\mathcal{C}$.

The global function of CA is a linear transformation, Φ: $C \to C$, which is used to determine the configuration at the next time step during the evolution of the CA, that is

$$C^{(T+1)} = \Phi(C^{(T)}).$$

If for a CA with bijective Φ, there exists another cellular automaton with global function Φ^{-1}, we call the former CA is *reversible*, and the latter CA is called its *inverse*. In such CAs the reverse evolution is possible [30,34]. The local transition function of a linear cellular automaton (LCA) with radius r takes the following form:

$$a_i^{(T+1)} = \sum_{j=-r}^{r} \alpha_j a_{i+j}^{(T)} (\mathrm{mod} q), \ 0 \le i \le N-1,$$

where $\alpha_j \in Z_q$. Since there are $2r+1$ neighbor cells for $< i >$, there exist $2r+1$ LCAs and each of them can be specified by an integer w called *rule number* which is defined as follows:

$$w = \sum_{j=-r}^{r} \alpha_j q^{r+j},$$

where $0 \le w \le q^{2r+1} - 1$.

The CAs mentioned so far are memoryless, i.e., the updated state of a cell depends on its neighborhood configuration only at the preceding time step. Nevertheless,one can consider cellular automata for which the states of neighboring cells at time T as well as $T-1, T-2, \ldots$ contribute to determine the state at time $T+1$. This type of CA is called the memory

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

cellular automata (MCA). A particular type of MCA called the tth order linear MCA(LMCA) , if its local transition function takes the following form:

$$a_i^{(T+1)} = f_1(\mathcal{N}_i^{(T)}) + f_2(\mathcal{N}_i^{(T-1)}) + \ldots + f_t(\mathcal{N}_i^{(N-t+1)}) (\mathrm{mod} q), 0 \le i \le N-1 \quad (4)$$

where f_i is the local transition function of a particular LCA with radius r, for $1 \le i \le t$. To start the evolution of a LMCA, t initial configurations $C^{(0)}, \ldots, C^{(t-1)}$ are required. what is more, to make this cellular automaton be able to evolve reversely, the following propositions [30,34] are needed.

Proposition 1. *If $f_t(\mathcal{N}_i^{(T-t+1)}) = a_i^{(T-t+1)}$, then the LMCA given by (Eq. 4) is reversible and its inverse is another LMCA with the following local transition function:*

$$a_i^{(T+1)} = \sum_{m=0}^{t-2} f_{t-m-1}(\mathcal{N}_i^{(T-m)}) + a_i^{(T-t+1)} (\mathrm{mod} q), \quad 0 \le i \le N-1,$$

In order to invert an LMCA of order t, exactly t configurations are required, this principle can be formally illustrated using the following proposition [30,34].

Proposition 2. *Let M be a tth order LMCA. Then, in order to compute $C^{(j+1)}$ for some $j \ge t$, exactly t configurations $C^{(j)}, C^{(j-1)}, \ldots, C^{(j-t+1)}$ are needed.*

3. Definitions of system model and security model for CAD-IP

The system model and security model of CAD-IP will be introduced in this section. A CAD-IP scheme consists of five entities as follows,

1. *KGC*: an entity to generate the system parameters and to setup the system.
2. *Owner*: an entity, which owns files to be stored on the public cloud servers, can be either an individual or an organization.
3. *Server*: an entity, which is operated by the cloud service provider, has high computing power, high performance and sufficient storage capacity. In our defined model, cloud servers are not required trusted.
4. *Verifier*: this entity is in charge of checking the availability and integrity of the stored files shares in cloud servers. By executing the defined-well interactive proof protocol with cloud servers, the verifier can make sure whether the files shares are intact as original.
5. *Combiner*: an entity, which collects the file shares from the cloud servers and recovers the original file, can be either an individual or an organization.

Definition 3 (Perfectness)**.** A CAD-IP scheme is called perfect if less than k file shares give no information about the original file, where k is the threshold of CAD-IP.

Definition 4 (CAD-IP)**.** A CAD-IP scheme is a tuple of five algorithms (Setup, ShareGen, TagGen, Proof, Recovery), which is described as follows,

1. **Setup**(1^λ): inputs a security parameter set λ, it outputs a system public parameter set *params*, LMCAs, and a secret parameter set $\overline{params}$.
2. **ShareGen**(F, *params*): inputs a public parameter set *params* and a file to be stored F, it outputs a share collection of the original file F.
3. **TagGen**($\{s_f\}, \overline{params}$): inputs a share block s_f, a secret parameter set $\overline{params}$, it outputs the tag t_g for s_f.
4. **Proof**($\mathcal{SRV}, \mathcal{VER}$): is an interactive process for a cloud server $\mathcal{SRV}$ and a verifier $\mathcal{VER}$. At last, $\mathcal{VER}$ outputs a bit 0, 1 to denote false or true about the verification to $\mathcal{SRV}$.
5. **FileRecov**($\{s_f\}, \overline{params}$)**:** inputs a collection of file shares $\{s_f\}$ and a secret parameter set $\overline{params}$, it outputs the original file F.

Definition 5 (Unforgeability)**.** A CAD-IP scheme is unforgeable if for any (probabilistic polynomial) adversary $\mathcal{A}$ (malicious cloud server) the probability that $\mathcal{A}$ wins the CAD-IP game with a challenger $\mathcal{C}$ on a collection of file shadow blocks is negligible. The CAD-IP game between $\mathcal{A}$ and the challenger $\mathcal{C}$ can be described as follows,

1. *Setup*: The challenger $\mathcal{C}$ runs Setup(1^k) to get ($params, \overline{params}$). It sends the public parameter set *params* to $\mathcal{A}$ and keeps $\overline{params}$ confidential.
2. *First-phase queries*: The adversary $\mathcal{A}$ adaptively issues TagGen queries to $\mathcal{C}$ as follows.

   For a share block tag query s_{ij} stored in the cloud server S_i, the challenger $\mathcal{C}$ calculates the tag t_{ij} and sends it back to $\mathcal{A}$. Let $< f_{ij}, t_{ij} >$ be the queried block-tag pair for index $j \in \mathcal{J}_1$, where $\mathcal{J}_1$ is a set of indices to indicate the block tags which have been queried in this stage.
3. *Challenge*: The challenger $\mathcal{C}$ makes a challenge *chal* which defines a ordered collection $\mathcal{J}_2 = \{j_1, j_2, \ldots, j_c\}$, where $\mathcal{J}_2 \nsubseteq \mathcal{J}_1$.

4. *Second-phase queries*: The adversary $\mathcal{A}$ can make queries adaptively as the first-phase, the only restriction is that $\{j_1, j_2, \ldots, j_c\} \nsubseteq \mathcal{J}_1 \cup \mathcal{J}_2$.
5. *Forge*: The adversary $\mathcal{A}$ outputs $\mathcal{V}_c$ for the challenge *chal*.

We say the adversary $\mathcal{A}$ wins the CAD-IP game if the response $\mathcal{V}_c$ can pass $\mathcal{C}$'s verification.

Definition 6 ((ρ, δ)-Secure)**.** A CAD-IP scheme is called (ρ, δ)-Secure [38] if the cloud server corrupted ρ fraction of the whole blocks, the probability that the corrupted blocks are detected is as least δ.

4. Construction of CAD-IP

The proposed CAD-IP consists stages System Setup, Shares Generation, Tag Generation, Integrity Proof and File Recovery as follows. Assume the confidential file to be stored is F, and there exist n cloud servers used to store F. Suppose F is split to m blocks and the length of each block is m', i.e. the block can be seen as a vector consisting of m' elements. The permissible threshold of recovery is k, that is to say no less than k cloud servers can collaborative recover the original file F.

4.1. *System setup*

In this stage, the KGC generates large primes, chooses Pseudo-Random Permutation and constructs LMCAs for the subsequent algorithms according to Algorithm 1.

Algorithm 1: Setup(1^λ).

Input: security parameters $\lambda_p, \lambda_q, \lambda_k$.
1: chooses $g \in Z_p$ with order q
2: selects two large primes p, q
3: chooses Pseudo-Random Permutation σ which maps$\{0,1\}^{\lambda_k} \times \{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$
5: computes $g^{\mathbf{r}}$
6: for $i \leftarrow 1$ to m'
7: randomly selects r_i in Z_q
8: $g_i := g^{r_i}$
9: selects r_{ca} in 1 to $\lfloor (m' - 1)/2 \rfloor$
10: for $i \leftarrow 1$ to $k - 1$
11: chooses random numbers ω_i in 1 to $q^{2r_{ac}+1} - 1$
12: constructs a reversible LMCA of order k
12.1: for $j \leftarrow 0$ to m'
12.2: $a_j^{(T+1)} = 0$
12.3 for $i \leftarrow 0$ to $k - 1$
12.4: $a_j^{(T+1)} = a_j^{(T+1)} + f_{\omega_j}(\mathcal{N}_j^{(T-i)}) + a_j^{(T-k+1)} \pmod q$
12.5: $a_j^{(T+1)} = a_j^{(T+1)} + a_j^{(T-k+1)} \pmod q$
/*where f_{ω_j} is the local transition function of LMCA of radius r with rule number ω_i*/
end

4.2. *Shares generation*

As the design of our proposed CAD-IP aims to provide a secure distributed storage solution with robustness, we employ threshold secret sharing to ensure confidentiality and robustness. Owing to the cellular automata has the property of nonlinearity and easy implementation on hardware, we use it to realize the distribution of file shares and did not turn to existing expensive public key based secret sharing schemes. To begin with construction of CA, we firstly split the original file F to m blocks $\{F_j (1 \le j \le m)\}$ as shown in Fig. 1, and then each block can be denoted using a vector as follows, $F_j = (a_{j,1}, a_{j,2}, \ldots, a_{j,m'})$ $(a_{j,i'} \in Z_q, 1 \le i' \le m')$. Since there are m blocks F_j of the original file F, each F_j will be used as a seed to generate the initial configuration for one evolution sequence $\{C_j^{(T)}\}$, then these m blocks produce m evolution sequences $\{C_j^{(T)}\}_{1 \le j \le m}$. After that, the last n configurations of each evolution sequence $\{C_j^{(q_j+i)} (1 \le i \le n)\}$ will be used as the shares of each original block F_i.

The owner generates file shares for the cloud servers by following Algorithm 2.

Then, the shares of original file blocks $\{F_j\}$ for each server S_i are $\{s_{ij}\}$, the owner has completed the preparation of the shares for the original file F.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

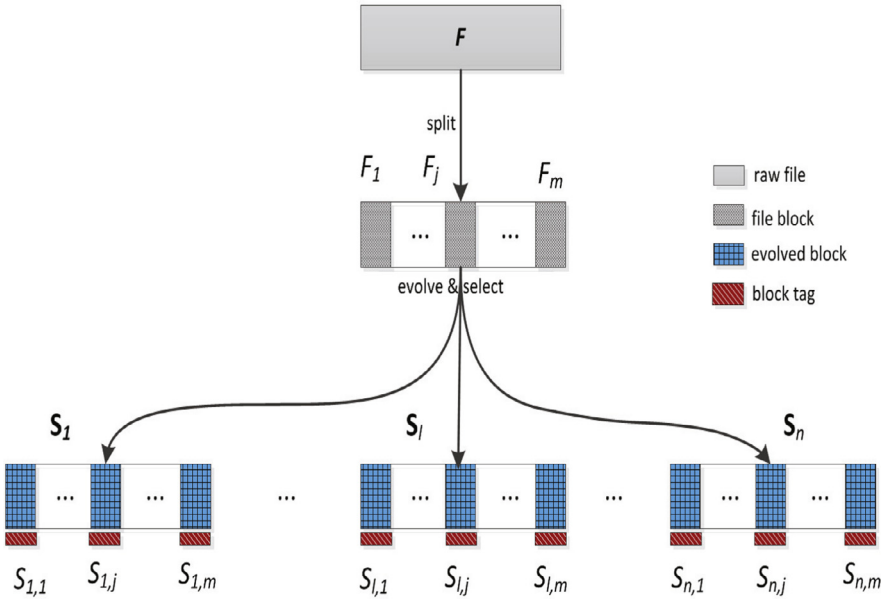

Fig. 1. File partition and distribution.

Algorithm 2: ShareGen(F, *params*).

input: $\{F_j(1 \le j \le m)\}$, *params*
1: for $j \leftarrow 1$ to m
2: $C_j^{(0)} := F_j$
3: for $i \leftarrow 1$ to $k-1$
4: generates $C_j^{(i)}$ using pseudo-random number generator
/* takes $C_j^{(0)}, C_j^{(1)}, \ldots, C_j^{(k-1)}$ as the initial configuration */
5: selects $q_j \ge k$ randomly
6: for $i \leftarrow 0$ to $(n+q_j)$
7: $C_j^{(k+i)}$=Evolve($C_j^{(k+i)}, C_j^{(k+i-1)}, \ldots, C_j^{(k+i-k)}$)/*evolve the LMCA*/
8: for $i \leftarrow 1$ to n
9: $s_{i,j} = C_j^{(q_j+i)}$.
end

4.3. *Tag generation*

To support checking the integrity of the whole share blocks in cloud servers, each share block would be appended with a short tag before it is uploaded to the server. The homomorphic hashing [15] is introduced to implement the efficient interactive possession proof. The file owner computes all tags for all shares by executing Algorithm 3.

Algorithm 3: TagGen($\{s_f\}$, $\overline{params}$).

input: $\{s_{i,j}\}$, *params*
1: for $i \leftarrow 1$ to n
2: for $j \leftarrow 1$ to m
3: $t_{i,j} = h(s_{i,j}) \bmod p$
end

After that, uploads $\{s_{ij},\ t_{ij}\}_{1 \le j \le m}$ to the cloud server S_i, and secretly sends $g^{\mathbf{r}} = \{g^{r_1}, g^{r_2}, \ldots, g^{r_{m'}}\}$ and $q_j(1 \le j \le m)$ to the verifier and combiner respectively.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

4.4. *Integrity proof*

To verify the integrity of the shares on the cloud servers, an interactive process will be executed between a cloud server and a verifier. The verifier can start the process periodically or arbitrarily. Without loss of generality, suppose the file shares on server S_l to be checked. The interactive proof between the verifier and cloud server is given by Algorithm 4.

Algorithm 4: Proof($\mathcal{SRV}, \mathcal{VER}$).

1: the verifier selects a secret e randomly and a challenge c
2: the verifier sends them to the cloud server S_l
3: the cloud server S_l computes V
3.1: V=$\vec{0}$
3.2: T=0
3.3: for $i \leftarrow 1$ to c
3.5: $V = V + s_{r_i} \bmod q$
3.6: $T = T + t_{l,r_i} \bmod p$ /*where $r_i = \sigma_e(i)$*/
3.7: sends (V, T) back to the verifier
4: the verifier checks h(V)=T holds or not using $g^{\mathbf{r}}$
4.1: if yes
4.2: believes that the shares are integral
4.3: else
4.4: believes that the integrity of the shares are damaged
end

During the proof, the cloud server only needs to perform few addition and multiplication operation, and the verifier is required to perform only one homomorphic hashing operation. Furthermore, the sampling based method is adopted to provide high efficiency, and pseudo-random permutation is used to ensure randomness and freshness, which can prevent potential replay attack.

4.5. *File recovery*

The recovery of the original file F needs the cooperation of some cloud servers. Owing to the property of threshold, just k cloud servers, not all n cloud servers are required to collaborate to finish the recovery. When to recover the file F from cloud servers, the combiner firstly randomly chooses l $(1 \le l \le n-k+1)$, next selects k servers $S_l, S_{l+1}, \ldots, S_{l+k-1}$ and downloads the corresponding shares $\tilde{S}_l, \tilde{S}_{l+1}, \ldots, \tilde{S}_{l+k-1}$, then the file F can be recovered by running Algorithm 5.

Algorithm 5: FileRecov($\{s_{i,j}, q_j\}$).

Input: shares $\{s_{i,j}, q_j\}$
1: for $j \leftarrow 1$ to m
2: for $i \leftarrow 0$ to $k-1$
3: $\tilde{C}_j^{(i)} = s_{l+k-1-i,j}$
4: for $i \leftarrow 1$ to $q_j + l - 1$
5: $\tilde{C}_j^{(k+i)} = \text{ReverseEvolving}(\tilde{C}_j^{(k+i-1)}, \tilde{C}_j^{(k+i-2)}, \ldots, \tilde{C}_j^{(k+i-k)})$
6: $F_j = \tilde{C}_j^{(k+i)}$
end

Now, the file F has been recovered since $F = (F_1, F_2, \ldots, F_m)$. Observed from the steps above, only q_j evolutions of LMCA are performed by the verifier. Since the evolution of LMCA can be easily implemented using hardware, the verifier can fast finish the verification.

5. Analysis

5.1. Correctness

As the fundamental requirements, correctness must be satisfied for a CAD-IP scheme. The illustration of correctness of our proposed CAD-IP scheme is given by the proof of Theorem 1.

Theorem 1. *The verifier can successfully check the integrity of the file shares in each cloud server according to the steps in 4.4.*

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

Proof. Without loss of generality, suppose the file shares in the cloud server S_l will be checked by the verifier. We know that the file shares stored in S_l are

$$\tilde{S}_l' = (s_{l,1}, s_{l,2}, \ldots, s_{l,m})$$

according to the definition in Section 4.2, where $s_{l,j} = C_j^{(q_j+l)} = (a_{j,1}^{(q_j+l)}, a_{j,2}^{(q_j+l)}, \ldots, a_{j,m'}^{(q_j+l)})'$, this means

$$\tilde{S}_l' = (C_1^{(q_j+l)}, C_2^{(q_j+l)}, \ldots, C_m^{(q_j+l)}) = \begin{pmatrix} a_{1,1}^{(q_j+l)} & a_{2,1}^{(q_j+l)} & \ldots & a_{m,1}^{(q_j+l)} \\ a_{1,2}^{(q_j+l)} & a_{2,2}^{(q_j+l)} & \ldots & a_{m,2}^{(q_j+l)} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1,m'}^{(q_j+l)} & a_{2,m'}^{(q_j+l)} & \ldots & a_{m,m'}^{(q_j+l)} \end{pmatrix},$$

then we have

$$C_{i'}^{(q_j+l)} + C_{j'}^{(q_j+l)} = (a_{i',1}^{(q_j+l)} + a_{j',1}^{(q_j+l)}, a_{i',2}^{(q_j+l)} + a_{j',2}^{(q_j+l)}, \ldots, a_{i',m'}^{(q_j+l)} + a_{j',m'}^{(q_j+l)})' \bmod q.$$

According to definition in 2.1, we have

$$h(C_{i'}^{(q_j+l)}) = \prod_{t=1}^{m'} g_t^{a_{t,i'}^{(q_j+l)}}.$$

So Eq. (5) holds

$$h(C_{i'}^{(q_j+l)} + C_{j'}^{(q_j+l)}) = \prod_{t=1}^{m'} g_t^{a_{t,i'}^{(q_j+l)} + a_{t,j'}^{(q_j+l)}} = \prod_{t=1}^{m'} g_t^{a_{t,i'}^{(q_j+l)}} \times \prod_{t=1}^{m'} g_t^{a_{t,j'}^{(q_j+l)}}, \tag{5}$$

that is

$$h(C_{i'}^{(q_j+l)} + C_{j'}^{(q_j+l)}) = h(C_{i'}^{(q_j+l)}) h(C_{j'}^{(q_j+l)}).$$

In other words, the equation $h(V) = T$ holds from the derivation above. □

5.2. Security

The goals of potential attacker $\mathcal{A}$ to CAD-IP can be categorized into two types, namely breaking confidentiality and integrity of the file shares on servers $S_l (1 \le l \le n)$. As for the former, the attacker $\mathcal{A}$ attempts to derive some information about F from the shares in some sever S_l, or to derive some information about F by colluding with multiple servers. As for the latter, $\mathcal{A}$ may delete or modify some share blocks in some server, even respond to the user with a forged pair $(\tilde{V}, \tilde{T})$. In this section, we illustrate the confidentiality, unforgeability, and (ρ, δ)-Secure) of our proposed CAD-IP scheme.

5.2.1. Confidentiality

Suppose the attacker $\mathcal{A}$ tries to derive some information about the file F from a single block $s_{l,j}$ which stored in server S_l, where $\mathcal{A}$ can be an outside attacker, even the server itself. However, as $s_{l,j} = C_j^{(q_j+l)}$ is evolved from $C_j^{(0)}$, $\mathcal{A}$ cannot evolve the cellular automata since he does not have other $k-1$ neighbor configurations, so that $\mathcal{A}$ will get nothing about $C_j^{(0)}$. If $\mathcal{A}$ attempts to choose $k-1$ neighbor configurations randomly, the success possibility of deriving $C_j^{(0)}$ for $\mathcal{A}$ is $P_j = ((1/q)^{m'})^{k-1}$, and the success possibility to recover F is $(1/q)^{m'(k-1)m}$, which is negligible. Next, we formally illustrate the confidentiality of the proposed CAD-IP by Theorem 2.

Theorem 2. *The proposed CAD-IP protocol is perfect.*

Proof. Assume an attacker $\mathcal{A}$ has corrupted up to $k-1$ cloud servers to derive some information about the original file F. Let M be a LMCA of order tth. In order to compute $\tilde{C}_t^{(j+1)}$ for some j, exactly k configurations $\tilde{C}_t^{(j)}, \tilde{C}_t^{(j-1)}, \ldots, \tilde{C}_t^{(j-k+1)}$ are needed. Suppose that the configuration $\tilde{C}_t^{(j-i)}$ is unknown for some $0 \le i \le k-1$. Let $\tilde{\Phi}_p$ be the global function of $\tilde{M}$. In order to get $\tilde{C}_t^{(j+i)}$, where

$$\tilde{C}_t^{(j+i)} = (a_{t,1}^{(j+1)}, a_{t,2}^{(j+1)}, \ldots, a_{t,m'}^{(j+1)}),$$

$\mathcal{A}$ has to conduct reverse evolution of M according to the following linear system [30,34]:

$$\tilde{C}_t^{(j+i)} = \sum_{p=0, p\neq i}^{k-1} \tilde{\Phi}_p(\tilde{C}_t^{(j-p)}) + \tilde{\Phi}_i(\tilde{C}_t^{(j-i)}) (\bmod q). \tag{6}$$

In Eq. (6), $\tilde{C}_t^{(j+i)}$ and $\tilde{C}_t^{(j-i)}$ are unknown, $\tilde{C}_t^{(j-p)} (0 \le p \le k-1, p \neq i)$ are known, and then Eq. (6) is a system of m' equations with $2m'$ unknowns. This means no information about $\tilde{C}_t^{(j+i)}$ can be derived. According to Definition 3, the proposed CAD-IP scheme is perfect. □

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

5.2.2. (ρ, δ)-Secure

To accurately evaluate the capability of detecting modification over the blocks of our proposed CAD-IP scheme, we give the proof of (ρ, δ)-Secure of our CAD-IP scheme as follows.

Theorem 3. *Assume m share block-tag pairs stored in each cloud server, θ block-tag pairs are modified and c block-tag pairs are challenged, the proposed CAD-IP protocol is $(\theta/m, 1-((m-\theta)/m)^c)-$secure.*

Proof. In our proposed CAD-IP scheme, the cloud servers are required to operate only on specified rows in each challenge-response protocol execution, that is conducting sampling on selected rows randomly according to the Pseudo-Random Permutation σ instead of all share blocks on some server. In this way, the computation overheads of cloud servers are significantly reduced, while high probability of detection of data corruption is maintained. Next, we evaluate the probability that the verifier successfully detects the modification or deletion on share blocks, and our method is similar to [33]. Without loss of generality, assume server S_l deletes θ blocks out of the n share blocks stored in server $\tilde{S}_l$. Let c be the number of different blocks asked for proof by the verifier in a challenge. Use X to denote a discrete random variable which presents the number of blocks chosen by the verifier that exactly match the blocks deleted by S_l. The probability that at least one of the blocks picked by the owner matches one of the blocks deleted by S_l, denoted as P_X, can be evaluated as follows [33].

$$\begin{aligned} P_X &= P\{X \geq 1\} \\ &= 1 - PX = 0 \\ &= 1 - \tfrac{(m-\theta)}{m}\tfrac{(m-1-\theta)}{m-1}\cdots\tfrac{(m-c+1-\theta)}{m-c+1}, \end{aligned}$$

then we have

$$1-\left(\frac{m-\theta}{m}\right)^c \leq P_X \leq 1-\left(\frac{m-c+1-\theta}{m-c+1}\right)^c.$$

□

However, P_X indicates the probability that, if the cloud server S_l deletes θ blocks of the file, then the verifier would detect the server's misbehavior after the proof about the challenged c blocks.

5.2.3. Unforgeability

The goal of the attack is to forge a pair (V, T) which satisfies $h(V) = T$. If an attacker $\mathcal{A}$ can find a collusion of the hash function h, then $\mathcal{A}$ succeeds. However, $\mathcal{A}$ cannot achieve it since a Probabilistic Polynomial Time(PPT) adversary cannot find a collusion. Next we illustrate it in detail. Assume the discrete log problem is hard over the group parameterized by (λ_p, λ_q), so that our scheme is secure.

Definition 7. A CAD-IP protocol is called unforgeable under an adaptive chosen-message attack if for any polynomial time adversary $\mathcal{A}$, if $\mathcal{A}$ can win the game defined in Definition 5 within time τ with the negligible probability $\epsilon(\tau)$.

Theorem 4. *Our proposed CAD-IP protocol scheme is unforgeable.*

Proof. We claim that if there exists a PPT adversary $\mathcal{A}$ can find a collision and win the game defined in Definition 5 within time τ with the probability $\varepsilon > \epsilon(\tau)$, then we can construct a challenger $\mathcal{C}$ to solve the discrete logarithm problem. To begin the construction, $\mathcal{C}$ is given the instance p, g, x, where $(g, x) \in G$, and its goal is to output δ such that $g^\delta = x$, where δ is unknown. $\mathcal{C}$ simulates the oracle to interact with the adversary $\mathcal{A}$ as follows.

Setup: the challenger $\mathcal{C}$ chooses $\beta_1, \beta_2, \ldots, \beta_{m'} \in \{0,1\}$ and $\mu_1, \mu_2, \ldots, \mu_{m'} \in \{0, 1, 2, \ldots, q-1\}$ randomly. For $i = 1, 2, \ldots, m'$, it sets $g^{\mathbf{r}} = (g_1, g_2, \ldots, g_{m'})$ as follows,

$$g_i = \begin{cases} g^{\mu_i} & \text{if } \beta_i = 0 \\ x^{\mu_i} & \text{else } \beta_i = 1 \end{cases}. \tag{7}$$

Let $< f_{ij}, t_{ij} >$ be the queried block-tag pair for index $j \in \mathcal{J}_1$, where $\mathcal{J}_1$ is a set of indices to indicate those block tags have been queried in this stage.

First-phase queries: The adversary $\mathcal{A}$ can adaptively issue TagGen queries to $\mathcal{C}$. Upon receiving the TagGen query of share block $\hat{S} = (\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_{m'})$ from $\mathcal{A}$, $\mathcal{C}$ computes $\hat{t}$ using $g^{\mathbf{r}}$ as follows,

$$\hat{t} = h(\hat{S}) = \prod_{i=1}^{m'} g_i^{\tilde{s}_i}$$

Challenge: The challenger $\mathcal{C}$ makes a challenge *chal* which defines an ordered collection $\mathcal{J}_2 = \{j_1, j_2, \ldots, j_c\}$, where $\mathcal{J}_2 \nsubseteq \mathcal{J}_1$.

Second-phase queries: The adversary $\mathcal{A}$ can make queries adaptively as the first-phase, the only restriction is that $\{j_1, j_2, \ldots, j_c\} \nsubseteq \mathcal{J}_1 \cup \mathcal{J}_2$.

Forge: The adversary $\mathcal{A}$ outputs $\mathcal{V}_c$ for the challge *chal*.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

However, if $\mathcal{A}$ can output $\mathcal{V}_c = (v_1, v_2, \ldots, v_{m'})$ successfully, he can generate another valid blocks $\mathcal{V}_c^* = (v_1^*, v_2^*, \ldots, v_{m'}^*)$ to produce a collision $H(\mathcal{V}_c) = H(\mathcal{V}_c^*)$, then $\mathcal{C}$ can solve the discrete logarithm problems as follows.

At first, $\mathcal{C}$ sets

$$a = \sum_{\beta_i=1} \mu_i (v_i - v_i^*) \bmod q. \tag{8}$$

then computes an inverse b of $a(\bmod q)$ using Euclid's algorithm, that is $ab \equiv 1(\bmod q)$. Computes

$$\delta = b \sum_{\beta_i=0} \mu_i (v_i^* - v_i) \bmod q. \tag{9}$$

Since $H(V) = H(V^*)$, we have

$$\prod_{i=1}^{m'} g_i^{v_i} = \prod_{i=1}^{m'} g_i^{v_i^*}.$$

substitute g_i with the corresponding value,

$$\prod_{\beta_i=0} g^{\mu_i v_i} \prod_{\beta_i=1} x^{\mu_i v_i} = \prod_{\beta_i=0} g^{\mu_i v_i^*} \prod_{\beta_i=1} x^{\mu_i v_i^*},$$

By re-arranging it, we get

$$\prod_{\beta_i=0} g^{\mu_i (v_i - v_i^*)} = \prod_{\beta_i=1} x^{\mu_i (v_i^* - v_i)},$$

that is

$$g^{\sum_{\beta_i=0} \mu_i (v_i - v_i^*)} = x^{\sum_{\beta_i=1} \mu_i (v_i^* - v_i)}.$$

According to Eq. (8), we then get

$$g^{\sum_{\beta_i=0} \mu_i (v_i - v_i^*)} = x^a.$$

Next, with raising both sides of the equation above to the power b, we have

$$(g^{\sum_{\beta_i=0} \mu_i (v_i - v_i^*)})^b = (x^a)^b$$

that is

$$g^{\delta} = x \bmod p.$$

At this moment, it means that $\mathcal{C}$ has obtained the solution δ for the discrete logarithm problem instance (g, x) with the probability $\varepsilon' > \varepsilon$, which contradicts the assumption. With Definition 7, we know that our proposed CAD-IP scheme is unforgeable. □

5.2.4. Comparison on security and functionality

In this section, we compare our scheme with existing schemes which concentrate on the distributed data storage or data sharing. The scheme in [32] investigates a mechanism to support on-the-fly verification for remote stored data based on homomorphic hashing, it also achieves robustness based on erasure-encoded techniques, however, it cannot prevent malicious servers to reveal the stored data. Although the schemes in [16,22] consider multiple data owners scenario of data storage in semi-trusted servers, and leverage attribute-based encryption (ABE) to ensure the confidentiality and access control of the data, it cannot detect the modification on the stored data and cannot support update dynamically on the data once they are uploaded to the servers. Furthermore, the encrypted data would be unrecoverable permanently once they are damaged. The scheme in [27] develops a mechanism to provide availability and suitability using the homomorphic token and erasure code for outsourced data, however, it cannot ensure confidentiality of the stored data since all the data are in plaintext. Although the scheme in [28] supports robust data storage using erasure code-based techniques, it cannot detect malicious modification over the stored data and cannot support dynamic update. The overall comparison of security and functionality with existing distributed cloud storage schemes is given in Table 1, however, only our proposed CAD-IP scheme achieves all security characteristic.

5.3. Performance

In this section, the computation and communication cost of the presented CAD-IP scheme are evaluated firstly, then the flexibility is discussed. Suppose the total number of share blocks stored in each server is m, and there exist c blocks will be verified in each challenge.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

Table 1
Comparison of security and function.

Scheme	Integrity	Confidentiality	Robustness	Dynamics
Li [16]	×	√	×	×
Liu [22]	×	√	×	×
Wang [27]	√	×	√	√
Lin [28]	×	√	√	×
Krohn [32]	√	×	√	×
CAD-IP	√	√	√	√

Table 2
Comparison of computation cost during proof.

Scheme	Server	Verifier
Ateniese [40]	cT_{pt}	$T_{de} + 2T_{pg}$
Zhu [37]	$(2cs + s + 3c)T_{ml} + (sc + c)T_{ex} + sT_{pr}$	$(c + s)T_{ml} + (c + s)T_{ex} + 3T_{pr}$
Wang [38]	$2cT_{ml} + 2cT_{ex} + 2cT_{pt}$	$(c + s)T_{ml} + (c + s)T_{ex} + 2cT_{pt} + 2T_{pr}$
CAD-IP	$cT_{ml} + cT_{pt}$	$m'T_{ml} + m'T_{ex}$

5.3.1. Computation

The computation overhead in Setup, ShareGen, TagGen, Proof, Recovery are evaluated in this section. Assume that the file F is split to m blocks and the length of each block is m', and the total number of the server is n, the permissible threshold number of the cloud servers to recover F is k. During setup, the computation overhead is $m'T_{ex}$ since $g_i = g^{r_i}(1 \le i \le m')$ need to be computed, where T_{ex} denotes the running time of single exponential operation. To confidentially store the file to the cloud servers, each file block should be converted to n shares using k cellular automata evolution, so that the overhead in this stage is mnT_{ev}, where T_{ev} denotes the running time of CA's single evolution. During the stage of TagGen, to generate a tag for each block, the owner needs to complete m' exponential operation, so the overhead in this stage is $mnm'T_{ex}$. As for the overhead of Proof, suppose the number of the challenged blocks is c, the server needs to complete c addition, random permutation and multiplication operation respectively. However, the addition and random permutation can be finished quickly, so the cost of these two operation can be neglected and the computation overhead for the server is $c(T_{ml} + T_{pt})$, where T_{ml} and T_{pt} represent the running time of single multiplication operation and Pseudo-Random Permutation respectively. To check the results from the cloud server, the verifier needs to complete m' exponent and multiplication operation respectively, so the overhead of the verifier is $m'(T_{ml} + T_{ex})$. When to recover the file, the combiner needs to collect the file shares from k cloud server firstly, then complete k CA's evolution operation, so the overhead for recovery is kT_{ev}. When the original file F needs to be recovered, the combiner should perform $\sum_{j=1}^{m}(q_j + l - 1)$ evolution operation, so that the computation cost in Recovery is $(\sum_{j=1}^{m}(q_j + l - 1))T_{ev}$.

However, the operations in Setup, ShareGen, TagGen and Recovery is one-off, so the efficiency in Proof stage should be given priority since the verifier can make unlimited challenges. Comparison with existing integrity proof schemes on computation cost during proof is conducted and the overall results are shown in Table 2, where T_{de}, T_{pr}, T_{pg} denote the running time of single decryption, pairing and pseudo-random generation operation respectively. The scheme in [40] employs symmetric encryption for verification, so its efficiency depends on the decryption of messages from the server. A proxy-like entity is utilized to assist the verifier to finish the verification in [37,38], for convenience, we take the proxy-like entity and the server together as server. As the bilinear pairing is used to construct the interactive proof, computation costs in [37,38] are much heavier than other schemes. It can be observed that our proposed CAD-IP gains much advantage on computational during the proof of integrity.

5.3.2. Communication

In the proposed CAD-IP scheme, only distribution of file shares and proof stage produce communication overhead. Since the file shares and corresponding tags are uploaded to the cloud servers once and for all, the communication overheads in this stage would not have influence on the performance of the whole system dramatically, hence we focus on the communication overheads from the Proof stage. As stated in Section 4.4, the process of interactive proof consists of challenge and response. When to issue a challenge, the verifier sends $\{e, c\}$ to the cloud server, so that the communication overhead during challenge is $\log_2 n + \log_2 q$, where n is the total number of cloud servers. Upon receiving the challenge, the cloud server will compute $\{V, T\}$ and respond it to the verifier, that is to say, the communication overhead of response stage is $(m'\log_2 q + |G|)$, where $|G|$ denotes the length of the element in group G. However, the operations in Setup, ShareGen, TagGen and Recovery is one-off, so the efficiency in Proof stage should be given priority since the verifier can make unlimited challenges. Similarly, comparison on communication cost during proof is conducted as well, and the concrete results are shown in Table 3, where l_λ denote the length of security parameter, and $|G_1|$, $|G_2|$ represent the length of elements in cyclic group G_1, G_2. From the comparison of performance,

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

Table 3
Comparison of computation cost during proof.

Scheme	Chal	Response
Ateniese [40]	$log_2 m + l_\lambda$	$l_\lambda + log_2 q$
Zhu [37]	$c(log_2 m + log_2 q)$	$\|G_1\| + \|G_2\| + s log_2 q$
Wang [38]	$log_2 m + 2 log_2 q$	$\|G_1\| + s log_2 q$
CAD-IP	$log_2 m + l_\lambda$	$m' log_2 q + log_2 p$

5.4. Flexibility

As a practical cloud storage system, dynamical data storage should be supported by the cloud service provider. In addition, the service should be universal. In this section, the flexibility of our proposed CAD-IP will be illustrated.

5.4.1. Scalability of verification

The verification of our proposed CAD-IP can be easily extended to the case of delegation and group-oriented situation. To enable the verification to be performed periodically when the verifier is out of office or the computation condition is not available, universal verification should be equipped. However, this can be achieved through distributing verification rights to multiple entities or delegating the verification rights to a proxy verifier. As for distribution of verification, the secret vector $g^{\mathbf{r}}$ can be divided to $|G_v|$ shares using (k, n)-threshold scheme [29], where $|G_v|$ denotes the amount of the group members, and each will be sent to every member of the verification group G_v. When to verify the integrity of the file shares, k members, not all group members required, can finish the verification collaboratively by recovering $g^{\mathbf{r}}$. In addition, the verifier can delegate the verification rights to another entity through delegation techniques [50], where the proxy verifier use a warranty generated by the original verifier to complete the verification of the integrity.

5.4.2. Dynamics

Since changes for storage is common, our scheme is designed for this requirement. The proposed CAD-IP scheme supports dynamic operations over the file F, such as modifying, appending, deleting. As for modifying F, assume the section of F to be modified is F_j, then the file owner generates the newly shares $\{s'_{1,j}, t'_{1,j}\}, \{s'_{2,j}, t'_{2,j}\}, \ldots, \{s'_{n,j}, t'_{n,j}\}$ and sends them to $S_1, S_2, \ldots, S_n$ respectively, then the server S_l replaces his block-tag pair $(s_{l,j}, t_{l,j})$ with $(s'_{l,j}, t'_{l,j})$. As for appending a new section F_{m+1} to F, the owner generates the shares $\{s_{1,m+1}, t_{1,m+1}\}, \{s_{2,m+1}, t_{2,m+1}\}, \ldots, \{s_{n,m+1}, t_{n,m+1}\}$ and uploads them to $S_1, S_2, \ldots, S_n$ respectively, then the server S_l appends $(s_{l,m+1}, t_{l,m+1})$ to the file share-tag pairs. If some of the section of F, assume F_j, to be deleted, the owner needs to notify server S_l to delete $\{s_{l,j}, t_{l,j}\}(1 \le l \le n)$. Note that these operations for each server just conducted over some single share block, not the entire block-shares. In this way, the performance of the system would be optimized since the communication overhead and computation overhead are reduced.

6. Conclusion

Existing storage efforts focus either on the confidentiality or integrity, however, the robustness is ignored in the process. Malicious attacks and natural disasters (earthquake, fire, etc.) may damage the stored data, though backup facilities are provided by the cloud service providers, because the stored data is encrypted by the same private key, once this key is compromised or destroyed, then the confidentiality cannot be ensured. To our knowledge, this work is the first comprehensive attempt to consider robustness, confidentiality, and integrity all together for data storage. We develop a novel distributed storage scheme for confidential data, which is a combination of cellular automata, threshold sharing and homomorphic hashing. In this way, our scheme can not only provide confidentiality and robustness, but also enable users to effectively detect any modification of deletion of stored data in cloud. Moreover, the proposed scheme can prevent forgery attacks as its security relies on CRHF family. Due to the dynamic characteristic of our scheme, it is convenient to update the file shares. Our proposed scheme aims to provide a secure storage solution for cloud, however, it can be applied to other distributed system as well. In our future work, more efficient file partition methods will be investigated to accommodate large file situations.

Acknowledgment

Our work was supported by the National Social Science Foundation of China (No. 14CTQ026).

References

[1] Mell P., Grance T. 2011. The NIST definition of cloud computing.
[2] Lounis A, Hadjidj A, Bouabdallah A, Challal Y. Secure and scalable cloud-based architecture for e-health wireless sensor networks. In: Computer communications and networks (ICCCN), 2012 21st international conference on. IEEE; 2012. p. 1–7.
[3] Mao Y, Wang J, Sheng B. Skyfiles: efficient and secure cloud-assisted file management for mobile devices. In: Communications (ICC), 2014 IEEE international conference on. IEEE; 2014. p. 4202–7.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

[4] Pettey C, Tudor B. Gartner says worldwide cloud services market to surpass $68 billion in 2010. Gartner Inc., Stamford, Press release; 2010.
[5] Rimal BP, Choi E, Lumb I. A taxonomy and survey of cloud computing systems. In: INC, IMS and IDC, 2009. NCM'09. Fifth international joint conference on, IEEE. IEEE; 2009. p. 44–51.
[6] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, et al. A view of cloud computing. In: Communications of the ACM, vol. 53(4); 2010. p. 50–8.
[7] Kandukuri BR, Paturi VR, Rakshit A. Cloud security issues. In: Services computing, 2009. SCC'09. IEEE international conference on. IEEE; 2009. p. 517–20.
[8] Chen Y, Paxson V, Katz RH. Whats new about cloud computing security. University of California; 2010. Berkeley Report No. UCB/EECS-2010-5 January vol. 20(2010). 2010-5.
[9] Mather T, Kumaraswamy S, Latif S. Cloud security and privacy: an enterprise perspective on risks and compliance. O'Reilly Media, Inc.; 2009.
[10] Ren K, Wang C, Wang Q. Security challenges for the public cloud. IEEE Internet Comput 2012;16(1):69–73.
[11] Hashizume K, Rosado DG, Fernández-Medina E, Fernandez EB. An analysis of security issues for cloud computing. J Internet Serv Appl 2013;4(1):1–13.
[12] Zissis D, Lekkas D. Addressing cloud computing security issues. Future Gener Comput Syst 2012;28(3):583–92.
[13] Yu S, Wang C, Ren K, Lou W. Achieving secure, scalable, and fine-grained data access control in cloud computing. In: INFOCOM, 2010 Proceedings IEEE; 2010. p. 1–9.
[14] Kamara S, Lauter K. Cryptographic cloud storage. In: Financial cryptography and data security. Springer Berlin Heidelberg; 2010. p. 136–49.
[15] López-Alt A, Tromer E, Vaikuntanathan V. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the forty-fourth annual ACM symposium on Theory of computing; 2012. p. 1219–34.
[16] Li M, Yu S, Zheng Y, Ren K, Lou W. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. In: Parallel and distributed systems, IEEE transactions on, vol. 24(1); 2013. p. 131–43.
[17] Wang G, Liu Q, Wu J. Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In: Proceedings of the 17th ACM conference on Computer and communications security; 2010. p. 735–7.
[18] Zhou L, Varadharajan V, Hitchens M. Achieving secure role-based access control on encrypted data in cloud storage. In: Information forensics and security, IEEE transactions on, vol. 8(12); 2013. p. 1947–60.
[19] Park N. Secure data access control scheme using type-based re-encryption in cloud environment. In: Semantic methods for knowledge management and communication. Springer Berlin Heidelberg; 2011. p. 319–27.
[20] Do JM, Song YJ, Park N. Attribute based proxy re-encryption for data confidentiality in cloud computing environments. In: Computers, networks, systems and industrial engineering (CNSI), 2011 First ACIS/JNU international conference on. IEEE; 2011. p. 248–51.
[21] Liu Q, Tan CC, Wu J, Wang G. Reliable re-encryption in unreliable clouds. In: Global telecommunications conference (GLOBECOM 2011), 2011. IEEE; 2011. p. 1–5.
[22] Liu Q, Wang G, Wu J. Time-based proxy re-encryption scheme for secure data sharing in a cloud environment. Inf Sci 2014;258:355–70.
[23] Xia Z, Wang X, Sun X, Wang Q. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans Parallel Distrib Syst 2015;27(2):340–52.
[24] Fu Z, Sun X, Liu Q, Zhou L, Shu J. Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEICE Trans Commun 2015;E98-B(1):190–200.
[25] Fu Z, Ren K, Shu J, Sun X, Huang F. Enabling personalized search over encrypted outsourced data with efficiency improvement. IEEE Trans Parallel Distrib Syst 2015. doi:10.1109/TPDS.2015.2506573.
[26] Fu Z, Wu X, Guan C, Sun X, Ren K. Towards efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement. IEEE Trans Inf Forensics Secur 2016. doi:10.1109/TIFS.2016.2596138.
[27] Wang C, Wang Q, Ren K, Cao N, Lou W. Toward secure and dependable storage services in cloud computing. Serv Comput IEEE Trans 2012;5(2):220–32.
[28] Lin HY, Tzeng WG. A secure erasure code-based cloud storage system with secure data forwarding. Parallel Distrib Syst IEEE Trans 2012;23(6):995–1003.
[29] Shamir A. How to share a secret. Commun ACM 1979;22(11):612–13.
[30] Wolfram S. Theory and applications of cellular automata, vol. 1. Singapore: World Scientific; 1986.
[31] Castro M, Liskov B. Practical byzantine fault tolerance. In: OSDI, Vol. 99; 1999. p. 173–86.
[32] Krohn MN, Freedman MJ, Mazieres D. On-the-fly verification of rateless erasure codes for efficient content distribution. In: Security and privacy, IEEE symposium on; 2004. p. 226–40.
[33] Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, et al. Provable data possession at untrusted stores. In: Proceedings of the 14th ACM conference on computer and communications security; 2007. p. 598–609.
[34] Eslami Z, Zarepour AJ. A verifiable multi-secret sharing scheme based on cellular automata. Inf Sci 2010;180. 2889C94
[35] Wang H, Zhang Y. On the knowledge soundness of a cooperative provable data possession scheme in multicloud storage. Parallel Distrib Syst IEEE Trans 2014;vol. 25(1):264–7.
[36] Wang H, Li J. Private certificate-based remote data integrity checking in public clouds. In: Computing and combinatorics. Springer International Publishing; 2015. p. 575–86.
[37] Zhu Y, Hu H, Ahn GJ, et al. Cooperative provable data possession for integrity verification in multicloud storage. Parallel Distrib Syst IEEE Trans 2012;vol. 23(12):2231–44.
[38] Wang H. Identity-based distributed provable data possession in multicloud storage. Serv Comput IEEE Trans 2015;8(2):328–40.
[39] Ateniese G, Burns R, Curtmola R, et al. Provable data possession at untrusted stores. In: Proceedings of the 14th ACM conference on computer and communications security. ACM; 2007. p. 598–609.
[40] Ateniese G, Di PR, Mancini LV, et al. Scalable and efficient provable data possession. In: Proceedings of the 4th international conference on Security and privacy in communication netowrks. ACM; 2008. p. 9.
[41] Wang H. Proxy provable data possession in public clouds. Serv Comput IEEE Trans 2013;6(4):551–9.
[42] Ren Y, Shen J, Wang J, Han J, Lee S. Mutual verifiable provable data auditing in public cloud storage. J Internet Technol 2015;16(2):317–23.
[43] Erway C.C., Küpçü A., Papamanthou C. et al. Dynamic provable data possession. ACM Transactions on Information and System Security (TISSEC); vol. 17(4). p. 15.
[44] Wu X, Sun W. Secret image sharing scheme with authentication and remedy abilities based on cellular automata and discrete wavelet transform. J Syst Softw 2013;86(4):1068–88.
[45] Zhou Y, Wang F, Qing S, et al. Dynamic multi-secret sharing scheme based on cellular automata. J Comput Res Dev 2012;49(9):1999–2004.
[46] Wang X, Luan D. A novel image encryption algorithm using chaos and reversible cellular automata. Commun Nonlinear Sci Numer Simul 2013;18(11):3075–85.
[47] Ping P, Xu F, Wang ZJ. Image encryption based on non-affine and balanced cellular automata. Signal Process 2014;105:419–29.
[48] Bakhshandeh A, Eslami Z. An authenticated image encryption scheme based on chaotic maps and memory cellular automata. Opt Lasers Eng 2013;51(6):665–73.
[49] Jin J. An image encryption based on elementary cellular automata. Opt Lasers Eng 2012;50(12):1836–43.
[50] Mambo M, Usuda K, Okamoto E. Proxy signatures for delegating signing operation. In: Proceedings of the 3rd ACM conference on computer and communications security. ACM; 1996. p. 48–57.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004

Yousheng Zhou received his Ph.D. from Beijing University of Posts and Telecommunications in 2011. He completed one-year postdoctorate work at Dublin City University, Ireland in 2016. He is currently an associate professor of the College of Computer Science and Technology, Chongqing University of Posts and Telecommunications. His research interests include network security and cloud security.

Feng Wang received his Ph.D. from Beijing University of Posts and Telecommunications in 2011. He is currently a lecturer of the College of mathematical sciences, De Zhou University. His research interests include secure authentication and secret sharing.

Fei Tang received his Ph.D. from the Institute of Information Engineering of Chinese Academy of Sciences in 2015. He is currently a lecturer of the College of Computer Science and Technology, Chongqing University of Posts and Telecommunications. His research interest is public key cryptography.

Xiaojun Wang is a senior lecturer of the School of Electronic Engineering of Dublin City University. He received his Ph.D. from the School of Engineering in Staffordshire University (then Staffordshire Polytechnic), England, UK in 1992. His research interests include information and network security, energy efficient ICT.

Please cite this article as: Y. Zhou et al., Cellular automata based secure distributed storage scheme with integrity proof, Computers and Electrical Engineering (2016), http://dx.doi.org/10.1016/j.compeleceng.2016.11.004