Energy-Efficient Algorithm for Assigning Verification Tasks in Cloud Storage

Guangwei Xu[1], Zhifeng Sun[1], Cairong Yan[1*], Xiujin Shi[1] and Yue Li[1]
[1] School of Computer Science and Technology, Donghua University
Shanghai, 201620, China
[e-mail: gwxu@dhu.edu.cn]
*Corresponding author: Cairong Yan

Received May 26, 2016; revised October 3, 2016; accepted November 15, 2016; published January 31, 2017

Abstract

Mobile Cloud Computing has become a promising computing platform. It moves users' data to the centralized large data centers for users' mobile devices to conveniently access. Since the data storage service may not be fully trusted, many public verification algorithms are proposed to check the data integrity. However, these algorithms hardly consider the huge computational burden for the verifiers with resource-constrained mobile devices to execute the verification tasks. We propose an energy-efficient algorithm for assigning verification tasks (EEAVT) to optimize the energy consumption and assign the verification tasks by elastic and customizable ways. The algorithm prioritizes verification tasks according to the expected finish time of the verification, and assigns the number of checked blocks referring to devices' residual energy and available operation time. Theoretical analysis and experiment evaluation show that our algorithm not only shortens the verification finish time, but also decreases energy consumption, thus improving the efficiency and reliability of the verification.

Keywords: Cloud storage, Data integrity verification, Energy-efficient, Assigning verification tasks

The work was supported by Key Laboratory of Ecology and Energy-saving Study of Dense Habitat (Tongji University), Ministry of Education, the Fundamental Research Funds for the Central Universities(No.2232015D3-29), the National Natural Science Foundation of China (Nos. 61070032 and 61300100), Shanghai Natural Science Foundation (Nos. 15ZR1400900 and 16ZR1401100), and Shanghai Education Scientific Research Project (No. C160076).

https://doi.org/10.3837/tiis.2017.01.001 **ISSN : 1976-7277**

1. Introduction

In recent years, Mobile Cloud Computing (MCC) has widely been used, which is a computing platform for mobile devices such as smart phones, laptops, etc. It is combined with cloud computing, mobile computing and wireless networks. With the rapid development of devices' computing power and available resources, more and more applications and services (such as games, Webchat, mobile office, version control systems, etc.) are transferred into this platform [1]. Mobile cloud storage which is a fundamental service of the MCC expends the storage space for the data owners (shorten to owner) to store their data, e.g., contacts, calendars, SMS, and Notepad. Hence, although the mobile device is damaged or lost, the data stored in cloud storage are still intact and available. Mobile owners easily recover their backup data from the remote cloud storage over wireless network. It greatly reduces the risk of data loss and corruption [2]. Although the mobile cloud storage brings benefits to owners, it still triggers the data integrity threat since owners lose control on their data. Also, an adversary storage service provider deliberately hides the data loss or corruption due to its hardware error and careless operation. It results in that mobile owners great worry about the data integrity in the mobile cloud storage. Thus, the data integrity verification in mobile cloud storage is attracting considerable attention [3]. Generally, third party verifiers are requested to execute the verification since either side of cloud storage provider (shorten to CSP) or owner could not guarantee to provide unbiased results of the verification.

In real world, cloud-based storage synchronization platforms such as Dropbox for Business and Sugarsync, Version Control Systems (VCS) such as Subversion and Concurrent Versions System, enable multiple team members to synchronously access and modify same files on cloud servers anywhere anytime [4]. With the wide utilization of smart phones and laptops, mobile devices will execute these data's integrity verification by collaborative operation before these data are accessed and modified. However, mobile devices acting as the verifiers in the MCC also causes some problems due to their limited energy and available operation time. For example, the verification isn't fully completed within the expiration periods due to mobile devices frequently entering or exiting. In this way, the reliability and efficiency of the data verification are seriously reduced.

The current verification schemes [5-13] mainly focus on the data integrity for the traditional mode of network access and neglect some potential problems under wireless network access. The verifiers seem incredible while they are undertaken by mobile devices, since their energy and available operation time are limited. Furthermore, these schemes are also incompetent in verification's flexibility and customizability in terms of the expected finish time of the verification related to data characteristics and importance. To meet these demands, we propose an energy-efficient algorithm for assigning verification tasks (EEAVT) to improve the efficiency and reliability of the data verification. Since an owners' verification request composes of huge volume of checked data, it should be divided into several verification tasks each with a certain amount of checked data blocks. Afterward, the algorithm chooses multiple mobile verifiers (called a verification group) to execute these verification tasks. Finally, each mobile verifier undertakes one verification task according to her residual energy and available operation time. In this way, the algorithm can optimize the finish time of the verification request and minimize the energy cost in data verification process while these energy-limited mobile devices act as the verifiers, and meanwhile achieve credible verification results.

We organize the paper as follows. In section 2, we briefly summarize current researches on the data integrity verification and task assignment. Section3 describes the system model, problems and design goals. Section4 describes the task assignment algorithm in detail. Section 5 analyzes the security of the algorithm. Section 6 evaluates the performance of the proposed algorithm by simulations. Finally, we conclude the contributions and present future extension in section 7.

2. Related Work

Many data verification schemes based on PDP (Provable Data Possession) or POR (Proof Of Retrievablity) in recent years have been proposed in [5-9]. These schemes have resolved some problems of the data verification in terms of public verification, dynamic data operation, blockless verification, privacy protection, batch verification, etc. Recently, Yu et al. [10] studied key-exposure resistance in cloud Storage Auditing. Wang et al. [11] improved PDP with identity-based distributed in multicloud storage. Yuan et al. [4] allowed multiple cloud users to modify owner's shared data with integrity assurance, and proposed an integrity auditing scheme for cloud data sharing services characterized by multiuser modification. Ismail et al. [12] adopted a game theoretical analysis to audit a cloud provider's compliance with data backup requirements. Li et al. [13] advised a POR scheme with resource-constrained devices in cloud computing. These schemes pay more attention to the accuracy of data verification, but seldom concern about the credibility and energy consumption while mobile devices act as the verifier. Moreover, they are indifferent to verification's flexibility and customizability.

In mobile cloud computing, mobile devices' operation is constrained by computing resources and energy. To save the energy, they utilize the task-offloading technology to execute the computation tasks. The basic idea of task-offloading is to move the computation load to cloud servers with rich resource and powerful energy. In this case, the mobile client is only responsible for the light computation tasks. Thus, the computation and storage resources of mobile devices are saved to achieve the energy efficiency. Kumar [14] designed a basic model of tasks offloading for mobile users to ensure energy efficiency in terms of the privacy protection, security, network reliability, and data transmission capacity. In order to realize energy-efficient tasks offloading, the designer first needs to decide which part of a program is put into the cloud. Shumao [15] proposed a dynamic $K+1$ division algorithm to meet the various constraints where a given application is divided into an indissoluble part and K decomposable parts. Wu [16] designed an offloading decision model in unavailable network environment. The model solves optimal power problems by programming method with graph partitioning technique and utilizes Bayesian decision to estimate the current network status. Also, it combines with the bees ABC method to find the optimal partition, and saves mobile devices' energy under unreliable network environment.

Cloud computing centers can reduce energy consumption by dynamic power management (DPM) such as shutting down some free servers temporarily [17], or dynamic voltage and frequency scaling (DVFS) such as appropriately reducing servers' performance [18]. Deng[19] proposed an energy-aware probabilistic scheduling approach to balance between optimizing scheduling length and saving energy in a time–energy–probability constrained multi-task uniprocessor system. In the approach, each task execution time follows a probability distribution. Kliazovich et al. [20] proposed a network-consciousness energy efficient data center task scheduling algorithm (DENS). It selects the most appropriate computation resources for each task to minimize the energy consumption of a data center. Subsequently,

Kliazovich et al. [21] also proposed a task scheduling algorithm e-STUB which equivalently treats tasks' transmission and computation to minimize their cost and balance tasks scheduling among several servers. Wang [22] proposed a nested optimization framework based on two-stage game in mobile cloud computing system. In the first stage, each mobile device determines the offloading task requests. In the second stage, cloud computing platform dynamically allocates resources to minimize power consumption and save energy according to tasks arrival rate while task service time is limited. Zhang [23] proposed an energy efficient transcoding task dispatching algorithm in a media streaming cloud. Considering the service delays and energy efficiency, the algorithm utilizes Lyapunov optimization method to satisfy the quality of service and minimize the transcoding energy cost. Also, Zhang [24] proposed a transcoding strategy of service energy efficient tasks-shedding for green cloud computing. With this strategy, delay-sensitive or delay-tolerate transcoding tasks are dispatched to the cloud server. In this way, the energy cost of mobile devices is minimized while the response delay time is guaranteed. Zhang [25] proposed a mobile cloud optimization framework based on random radio channel energy. The framework dynamically configures CPU frequency and changes the data transmission rate to decide which mobile application is preferred to be executed at the local or in the cloud, and thus saves mobile devices' energy.

3. Models and problem statement

3.1 System model and assumptions

In the data verification model, there are three entities, i.e., data owner (DO), cloud service provider (CSP), a group of verifiers (VG), as shown in **Fig. 1**. The DO uploads his data and corresponding verification tags to CSP. Also, the DO easily accesses these data via the Internet after she signs a deal with the CSP for the outsourced data service. The VG is responsible for checking these data integrity and reporting the results of data integrity or corruption while the DO requests the data verification. In the model, we assume: 1) CSP communicates with VG over a secure channel. 2) Verifiers are not absolutely credible since their operation time and energy are limited. 3) The number of verifiers in one VG is large enough to meet owners' elastic and customizable verification request.

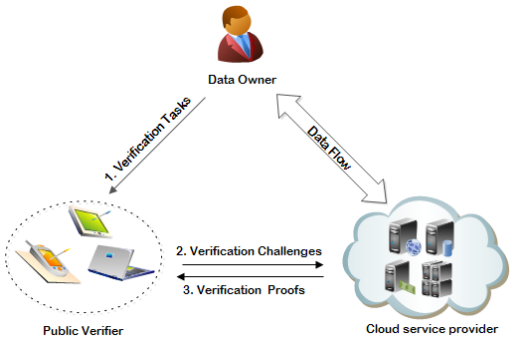


Fig. 1. The verification model of multiple mobile verifiers.

The verification process comprises three phases, i.e., setup, assigning verification tasks, and challenge-response. In setup phase, DO preprocesses data to generate public and private keys, initial parameters, verification tag of each block, and finally uploads the data and corresponding tags into CSP. In the phase of assigning verification tasks, each verifier in VG responses owners' verification request and is assigned a verification task according to the

limited task finish time, which includes a certain amount of checked blocks. In challenge-response phase, each verifier in VG independently challenges the suspicious data which consists of some data blocks' indices and the corresponding random number of each block, and then sends them to CSP. After receiving the proofs of data integrity from CSP, VG verifies them to judge whether these checked data are intact or corrupted.

DO's data are generally divided into many logical blocks and stored in CSP's servers. We utilize a four-tuple to represent owners' verification request, i.e., $VR = (M, n, T, P)$, where M is the checked data, n is the number of blocks in the checked data, T is the expected finish time, and P is the probability that the verification algorithm can accurately judge the verification results, generally $P \geq 95\%$. Let $VT = (c, t)$ be the verification task, which c represents the number of checked blocks in the verification task, and t is the finish time of the verification task. Obviously, c and t control the energy consumption of verification tasks. Generally speaking, if t is given, the more c is, the greater the energy cost of the verification tasks is. If c is a constant, the energy cost of verification tasks increases with the decrease of t.

The model of verification tasks assignment is shown in **Fig. 2**, where the verification request VR is divided into S verification tasks which are assigned to S verifiers. M is divided into n blocks, and the VG includes S ($S > 0$ and far less than n) verifiers. Let the number of checked blocks in the verification request be q and the ith ($i \in [1, S]$) verifier be assigned n_i blocks. If the ith verifier is assigned 0 block in the verification task, we let $n_i = 0$. Let the verification finish time of n_i blocks be t_i. We have $q \leq \sum_{i=1}^{S} n_i$ and $max(t_1, t_2, \cdots, t_S) \leq T$. Only if the total number of blocks $\sum_{i=1}^{S} n_i$ that is cumulatively checked by S verifiers is greater than or equal to the number of blocks in the verification request q, the verification request can be fully completed. Moreover, $max(t_1, t_2, \cdots, t_S)$ means that the maximum value in these data (i.e., $t_1, t_2, \cdots, t_S$) is taken, and simultaneously the maximum value must be less than or equal to the expected finish time T. Otherwise, the verification request cannot be completed within the expected finish time.

On the other hand, for different verification requests, CSP computes and responds the proofs of data integrity within a specified response time (RT). Assume L servers in CSP execute the verification computation. The operation frequency of the jth ($j \in [1, L]$) server is f_j ($f_1 \leq f_2 \leq \cdots \leq f_L$), which is corresponding to the server's response level of the verification RT_j. Let RT_1 be the lowest response level, and RT_L be the highest response level. The lowest response level RT_1 corresponds to the lowest operation frequency of the server's CPU, and represents that the response time of server computing the proof is the longest. In this case, the energy consumption of the server is the minimum since its CPU operates at the lowest frequency. Similarly, the highest response level RT_L corresponds to the highest operation frequency of the server's CPU, and indicates that the response time of server computing the proof is the shortest, and the corresponding server costs the maximum energy as its CPU operates at the highest frequency. Let the urgency level of a verification request be U, and $U = j$. The urgency level $U = j$ indicats that the server deals with the verification request with the response level RT_j. As the cloud computing has the high-performance computational capacity, the verification requests can be parallelly processed while different verifiers execute the verification tasks. Certainly, each verification request corresponding to a specific response level has different delay time.

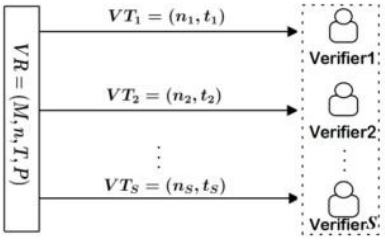


Fig. 2. The model of assigning verification tasks.

3.2 Problem statement

For a given verification request, it can be executed by one or several verfiers. In traditional verification, the energy-efficient strategy isn't considered since the permanently powered devices undertake these verification tasks. A simple method is to utilize the consistent hashing to assign the verification tasks according to verifier's capacity. However, this method doesn't take into account the characteristic of mobile devices and owners' elastic verification needs. The mobile devices have the limited energy and operation time. A mobile verifier is assigned the verification task which cannot go beyond the verifier's capability (i.e., energy and operation time). Otherwise, it not only wastes the verifier's computation resources and energy, but also cannot finish owners' verification requests.

Although mobile verifiers are requested to execute the verification tasks, they may not be able to return the verification results within the appointed time since these assigned verification tasks aren't completed due to mobile devices' limited energy and operation time. It greatly decreases the verification efficiency due to the verification failure or unreliable verification results. For example, one verifier is performing one verification task of 1000 blocks. If her residual energy and limited operation time aren't enough to complete the verification task of all these blocks, the verification will fail since the batch of data (1000 blocks) isn't fully verified. It not only wastes the verifier's computation resources, but also causes the unreliable verification results due to the incomplete verification task.

Moreover, if the verification task is urgent, it must be completed within the expected time. For example, a task of 1000 blocks needs to be finished within 5 seconds. Although a verifier has enough operation time, she cannot finish the task within the expected time due to her poor computation capacity. In the end, the verification fails due to the expire time of the verification task. Thus, an allocated verification task cannot go beyond the verifier's capability.

To ensure the verification efficiency and reliability, the verification request is divided into several verification tasks, and each task is assigned to one mobile verifier. It not only reduces energy waste (including the servers and verifiers), but also meets owners' elastic verification request. As both the residual energy and the available operation time of verifiers are different, they should be assigned the appropriate verification tasks. Certainly, if a verifier is assigned a small number of data blocks which are much less than his capability, his computation resources cannot be fully utilized.

3.3 Design goals

To save energy cost, the algorithm of assigning verification task should achieve the following goals:

1) The lower failure rate of verification tasks, i.e., more efficient and more reliable verification;

2) The elastic verification requests such as the detection rate and the expected finish time of

the verification;

3) Awareness of energy cost, i.e., verification tasks are assigned according to verifiers' residual energy and available operation time.

4. Algorithm of Assigning Verification Tasks

4.1 Overview and Symbol Definitions

To minimize the energy cost and improve the efficiency and reliability of the data verification, we design an algorithm EEAVT as shown in **Fig. 3**. To conveniently describe it, we define some notations in **Table 1**.

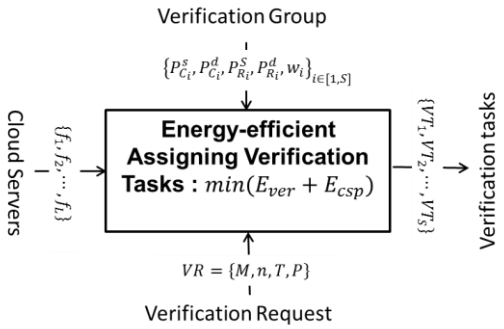


Fig. 3. The overview of EEAVT.

Table 1. Notations in algorithm

Notations	Description
w_i	Available operation time of the ith verifier
P_{Ci}^s	CPU power of the ith verifier in idle state
P_{Ci}^d	CPU delta power of the ith verifier in active state
P_{Ri}^s	Radio transmission power of the ith verifier in idle state
P_{Ri}^d	Radio transmission power of the ith verifier in active state
$E_{Ci}(.)$	Computation energy cost of the ith verifier
$E_{Ri}(.)$	Transmission energy cost of the ith verifier
$E_i^{ver}(.)$	Verification energy cost of the ith verifier
$E_j^{csp}(.)$	Computation energy cost of the jth server

4.2 Energy Consumption

The energy cost of mobile verifier mainly consists of CPU energy cost (i.e., verification computation) and network transmission energy cost (i.e., the transmission of verification challenges and proofs). Both the two energy costs are further divided into dynamic and static costs [18]. Let CPU power in idle state (i.e., static power) be P_C^S, and CPU power in active state (i.e. active power) be P_C^d. The CPU energy cost of the verifier $E_C(n)$ is expressed as

$$E_C(n) = P_C^S \cdot t(n) + P_C^d \cdot t^{ver}(n)\,, \qquad (1)$$

where $t^{ver}(n)$ is the proof verification time of n blocks, $t^{tran}(n)$ is the proof transmission

time of n blocks, $t^{csp}(n)$ is the proof generation time of n blocks, and $t(n)$ is the sum of $t^{ver}(n)$, $t^{tran}(n)$ and $t^{csp}(n)$. Similarly, let the ratio of transmission powers in idle state and active state be P_R^s and P_R^d respectively. The transmission energy cost of the verifier is

$$E_R(n) = P_R^s \cdot t(n) + P_R^d \cdot t^{tran}(n)\,. \qquad (2)$$

We get the total energy of n checked blocks for the verifier by

$$E^{ver}(n) = E_C(n) + E_R(n). \qquad (3)$$

The parameters of the ith verifier in VG are given by $VP_i = (P_{Ci}^s,\ P_{Ci}^d,\ P_{Ri}^s, P_{Ri}^d, w_i)$, where $P_{C1}^s \le P_{C2}^s \le \cdots \le P_{CS}^s$, $P_{C1}^d \le P_{C2}^d \le \cdots \le P_{CS}^d$. The energy cost of the ith verifier is $E_i^{ver}(n_i) = E_{Ci}(n_i) + E_{Ri}(n_i)$ by applying the equation (3). The total energy cost of S verifiers is $E_{ver} = \sum_{i=1}^{S} E_i^{ver}(n_i)$.

On the other hand, the main energy cost of the servers is the computation load of CPU operating the calculation tasks [21-24]. Thus, we focus on the energy cost of the servers generating the proofs in the verification. Assume that the CPU fundamental operatation frequency of each server is equal to f, which supports the server to maintain basic operations. The energy that the server generates the proofs of n blocks costs

$$E^{csp}(n) = \kappa t^{csp}(n) f^{\alpha}\,, \qquad (4)$$

where κf^{α} is the convex function of CPU operating frequency, and it generally sets $\alpha = 3$ and $\kappa = 1$ referring to [21-24]. Let the urgency level of verification request be $U = j$. Applying the equation (4), the computation energy cost of the server is

$$E_{csp} = \sum_{i=1}^{S} E_j^{csp}(n_i) = E_j^{csp}(n). \qquad (5)$$

Our objective is to minimize the energy cost in the verification while verifiers' service time (or energy) is limited and owners' expected finish time is given. Thus, the problem is optimized in mathematics by

$$Min(E_{ver} + E_{csp}) \quad s.t. \quad \begin{cases} t_i(n_i) \le T, & i \in [1,S] \\ w_i \ge t_i(n_i) \ge 0, & i \in [1,S] \end{cases}, \qquad (6)$$

where the time of the ith verifier checking n_i blocks is $t_i(n_i)$. $t_i(n_i) \le T$ represents the finish time of verification task within the expected finish time. $w_i \ge t_i(n_i) \ge 0$ indicates that

the verifier completes the assigned verification task during her available operation time. Specifically, the verification tasks $(n_1, n_2, \cdots, n_S)$ are assigned by

$$\begin{aligned} & Min(E_{ver} + E_{csp}) \\ & s.t. \quad \begin{cases} n_1 + n_2 + \cdots + n_S \geq q \\ n_i \geq 0, \ \forall i \in [1, S] \\ n_i \leq min\left[n_{w_i}, n_T, n\right], \forall i \in [1, S] \end{cases}, \end{aligned} \quad (7)$$

where n_{w_i} and n_T are the maximum number of assigned blocks in the task within the available operation time of the verifier and the expected finish time of the owner respectively.

4.3 Verification Time

To ensure that each verifier finishes the allocated verification task in device's operation time, we need to estimate verification finish time of checked data blocks, and avoid the verification failure due to the excessive verification tasks. In Section 4.2, the verification time consists of three parts, i.e., proof generation time, proof verification time and proof transmission time. All these time has a linear relationship with the number of checked data blocks. Assume that the network bandwidth is stable in the verification process.

For a given verification algorithm and network bandwidth B, the transmission time is $t^{tran}(x) = d(x)/B$, where $d(x)$ is a function of x checked blocks, and indicates the transmission capacity in the verification process. In addition, we provide a parameter as a baseline of verifiers' and servers' capacities. Let the average energy consumption of CPU in idle state and active state be P_{CO}^{s} and P_{CO}^{d} respectively, and the average running frequency of server's CPU be f_0. We utilize the linear regression analysis to estimate the proof generation time $t_0^{csp}(x)$ and the proof verification time $t_0^{ver}(x)$. Given the observation samples $\left\{(x_i, t_0^{csp}(x_i)\right\}_{1 \leq i \leq s}$, the theoretical regression of the proof verification time and checked blocks is represented as $t_0^{csp}(x) = ax + b + \varepsilon$, where a, b and ε are unknown. Let $\hat{a}$ and $\hat{b}$ be estimated values of a and b respectively. The estimated regression is expressed as $\hat{t}_0^{csp}(x) = \hat{a}x + \hat{b}$. Reffering to the least squares estimation, we let regression coefficients be $\hat{a} = \bar{t}_0^{csp}(x_i) - \hat{b}\bar{x}$ and $\hat{b} = (\sum_{i=1}^{s} x_i t_0(x_i) - s\bar{x}\bar{t}_0^{csp}(x_i))/(\sum_{i=1}^{s} x_i^2 - s\bar{x}^2)$, where $\bar{x} = \frac{1}{s}\sum_{i=1}^{s} x_i$ and $\bar{t}_0^{csp}(x) = \frac{1}{s}\sum_{i=1}^{s} t_0^{csp}(x_i)$. Correspondingly, the correlation coefficient and regression deviation are $r = \frac{\sum_{i=1}^{s}(x_i - \bar{x})\left(t_0(x_i) - \bar{t}_0^{csp}(x_i)\right)}{\sqrt{\sum_{i=1}^{s}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{s}\left(t_0(x_i) - \bar{t}_0^{csp}(x_i)\right)^2}}$, $\frac{\Delta b}{b} = \sqrt{\frac{\frac{1}{r^2} - 1}{n-2}}$ and $\frac{\Delta a}{\Delta b} = \sqrt{\overline{x^2}}$ respectively. Similarly, we also have $\hat{t}_0^{ver}(x) = \hat{c}x + \hat{d}$. Thus, the verification time of the ith verifier checking x blocks is computed by

$$\hat{t}_i(x) = \frac{f_0 \hat{t}_0^{csp}(x)}{f_i} + \frac{(P_{CO}^{s} + P_{CO}^{d})\hat{t}_0^{ver}(x)}{P_{Ci}^{S} + P_{Ci}^{d}} + t^{tran}(x), \quad (8)$$

while the operatation frequency of the server is f_j.

4.4 Algorithm Description

Multiple verifies checking the data integrity can greatly improve the efficiency and reliability of the verification. However, each verifier's verification operation time and computation capacity are different. To reduce energy consumption in the verification, we utilize EEAVT that the verification request is divided into several different verification tasks, and each verifier is assigned the appropriate number of verification tasks.

Let the minimum finish time (i.e., verification capability) be dw. The expected finish time of the owner T should be greater than dw to avoid beyond verifier's capability. Let the maximum finish time be t_m. Also, T should be less than t_m to take full advantage of a verification group. The energy awareness algorithm of assigning blocks is described as follows.

1) $T \geq t_m$ and $t_m \leq w_1$ mean that the finish time of verification request is very long and the verification operation time of the verifier $Ver1$ is long enough to meet the verification request. According to $E_1^{ver}(n) \leq E_2^{ver}(n) \leq \cdots \leq E_S^{ver}(n)$, $Ver1$ is able to complete the verification with the minimum energy consumption while the data block assignment scheme is $(n_1, n_2, \cdots, n_S) = (n, 0, \cdots, 0)$;

2) $T \geq t_m$ and $t_m > w_1$ mean that the owner's expected finish time is long enough. $Ver1$ has insufficient operation time to complete the verification request. Thus, it needs multiple verifiers. We have $w_i \geq \hat{t}_i(x) = g_i^{-1}(x)$ by applying the equation (8). The assigned data blocks for $Ver1$ satisfies $n_1 = n_{w_1} = \lfloor g_1(w_1) \rfloor$, where $\lfloor . \rfloor$ denotes the least integer greater than or equal to the current value. For the ith verifier ($i \geq 2$), the number of assigned blocks is computed by

$$n_i = \begin{cases} n_{w_i}, & if\ R \geq n_{w_i} \\ R\ , & if\ n_{w_i} > R > 0, \\ 0\ , & if\ R \leq 0 \end{cases}$$

where $R = n - \sum_{k=1}^{i-1} n_{w_k}$, $n_{w_i} = \lfloor g_i(w_i) \rfloor$.

3) $T \in (dw, t_m]$ means that the owner's verification request is very urgent, and it needs multiple verifiers to execute the verification. The analysis is similar to 2).

4) $T \leq dw$ means that owner's verification request is too harsh to exceed the maximum capacity of the verification group. In this case, it either changes owner's request or increases the number of verifiers in VG.

The process of algorithm is shown in Algorithm 1.

Algorithm 1: Energy Efficient-based Assigning Tasks

Input: $\{VP_i\}_{1 \leq i \leq S}, VR, U$
Output: $\{n_1, n_2, \cdots, n_S\}$
1: **Initialize** the blocks assigned to the ith verifier n_i with *0* and the unassigned blocks R with n.
2: $i \leftarrow 1$
3: **repeat**
4: **if** $w_i > T$ **then** $w_i \leftarrow T$ //the ith verifier finishs task
5: $n_i^* \leftarrow \lfloor g(w_i) \rfloor$ //the maximum verifiable blocks of the ith verifier
6: **if** $R - n_i^* > 0$
7: **then** $n_i \leftarrow n_i^*$
8: **else** $n_i \leftarrow R$
9: $R \leftarrow R - n_i^*$
10: $i \leftarrow i + 1$
11: **until** $i = S$ && $remain \leq 0$

Here, lines 1-2 initialize the verification parameters and counter. Lines 4-5 preprocess the service time of the ith verifier and compute the maximum verifiable blocks. Lines 6-11 assign the appropriate number of checked blocks until no block is left.

5. Algorithm Analysis

5.1 Verification Security

The proposed algorithm should ensure the verification correctness, resiting verification spoofing attacks, and preserving data privacy. Our algorithm is based on the homomorphic verification technology, which has been proved that it can ensure verification correctness, security, and preserving data privacy by many researches [7, 8]. Due to space limitation, we don't discuss these issues in this paper.

5.2 Verification Credibility

The credibility of the algorithm not only minimizes the energy cost, but also achieves effective verification results. Given the verification request $VR = (M, n, T, P)$, and e corrupted blocks in n blocks. The corrupted rate of stored data is $\rho_e = e/n$. Let the number of checked blocks be c. Thus, the detection rate of corrupted data is $P \geq 1 - (1 - \rho_e)^c$. For example, let $n = 10000$ and $e = 100$, it has $c \geq 300$, $c \geq 460$, and $c \geq 9901$ respectively while $P \geq 95\%$, $P \geq 99\%$, and $P = 100\%$. Generally speaking, the number of checked blocks is large enough while $P \geq 99\%$. Moreover, it enable the verification request to be finished by choosing the response level of verification RT. Finally, multiple verifiers further ensure that the verification tasks can be fully completed. Thus, credible verification results can be achieved.

5.2 Verification Cost

There are several methods to minimize the energy cost. The first one is to choose the appropriate number of checked blocks since the more number of checked blocks costs much energy. The second one is to adjust the expected finish time of verification according to the corresponding response level RT since more urgent verification request costs more energy in the verification. The last one is to optimize the number of verifiers in VG since more verifiers spends more data transmission energy.

Assume several valid verifiers in VG who can fully complete the allocated verification tasks execute the verification. For example, there are n blocks and S verifiers. If the expected verification finish time T is long enough for one verifier to finish the verification request, she can perform the verification task of all n blocks alone. VG's energy consumption is $E_1^{ver}(n)$ while all blocks are verified by $Ver1$. In this case, the energy cost is the lowest. If the expected finish time T is too short for one verifier to finish the request, S verifiers cooperate on executing the verification, i.e., each verifier averagely performs n/S blocks. In this case, we have $T = t^{ver} + t^{tran} + t^{csp}$, $t^{ver} \in (t_S^{ver}\left(\frac{n}{S}\right), t_1^{ver}(n)]$, $t^{tran} \in (\frac{d\left(\frac{n}{S}\right)}{B}, \frac{d(n)}{B})$ and $t^{csp} \in (t_L^{csp}\left(\frac{n}{S}\right), t_1^{ver}(n)]$. The range of verification time is $(dw, t_m] = (t_S^{ver}\left(\frac{n}{S}\right) + t^{tran}\left(\frac{n}{S}\right) + t_L^{csp}\left(\frac{n}{S}\right), t_1^{ver}(n) + t^{tran}(n) + t_1^{csp}(n))$, and $T > dw$. VG's energy consumption is

$S \cdot E_S^{ver}\left(\frac{n}{S}\right)$ while all blocks are verified by S verifiers. In this case, the energy cost is the highest.

6. Experiment Evaluation

In this section, we evaluate the performance of our algorithm. We choose two servers each with Dell PowerEdge R420 (Intel Xeon E5-2403 1.8GHz CPU and 8GB memory) and one disk array cabinet with IBMDS3300 (4×1T hard disks) to build a cloud storage platform and act as CSP. Verifiers execute the verification on several laptops each equipped with the Intel Core Duo 2.4GHz CPU and 4GB memory. The algorithm utilizes Java Pairing-based cryptography library. In experiments, to well evaluate experiments, we let the size of test file be 4MB, 40MB, and 400MB respectively, and each block length be 4KB, 8KB, and 16 KB respectively. For example, the number of blocks is 10,000 while the size of file is 40MB, and each block length is 4KB. let $S = 10$, $B = 200$ kbps, $L = 5$, $\kappa = 1$, $\alpha = 3$, $f_0 = 2.6$, $\{f_j\} = 1.8{:}\ 0.3{:}\ 3.0$, $P_{C0}^s = 2.5$, $P_{C0}^d = 4.5$, $P_{R0}^s = 1.25$, $P_{R0}^d = 1.25$, $\{P_{Ci}^s\} = 2.0{:}\ 0.2{:}\ 3.8$, $\{P_{Ci}^s + P_{Ci}^d\} = 6{:}\ 0.4{:}\ 9.6$, $\{P_{Ri}^s\} = 1{:}\ 0.05{:}\ 1.5$, and $\{P_{Ri}^d\} = 1{:}\ 0.0.5{:}\ 1.5$ [16]. The test results are the average value of 20 tests.

To evaluate the performance of our algorithm, we compare our algorithm EEAVT with other representative algorithms such as the algorithm of [25] defined as RRAVT, and the algorithm of [12] defined as SAVT.

6.1 Expected Verification Time

Let $s = 100$. To evaluate the verification time, we set correlation parameters of $\hat{t}_0^{ver}(x) = \hat{c}x + \hat{d}$ and $\hat{t}_0^{csp}(x) = \hat{a}x + \hat{b}$, as shown in **Table 2** and **Table 3**. The estimated proof verification time $\hat{t}_0^{ver}$ and proof generation time $\hat{t}_0^{csp}$ have nearly linear correlation with the number of checked blocks, and the deviation can be ignored. Therefore, the linear function gives a good description of the relationship between the number of checked blocks and verification finish time.

Table 2. Parameters of proof generation

$\hat{a}$	$\hat{b}$	r	$\Delta b/b$	$\Delta a/\Delta b$
0.014	0.6255	0.99956	0.0242	4.12E-05

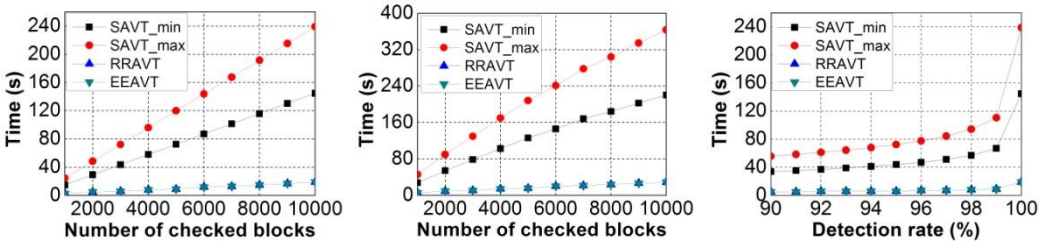
Table 3. Parameters of proof verification

$\hat{c}$	$\hat{d}$	r	$\Delta c/c$	$\Delta c/\Delta d$
0.0028	0.055	0.99947	0.0053	9.19E-06

6.2 Verification Finish Time

The verification finish time is the duration of one verification group finishing a given verification request, which includes a certain amount of checked blocks. Obviously, if the number of blocks and the detection rate are given, the stronger the capacity of the verification group is, the shorter the verification finish time is. Let $q = 1000{:}\ 100{:}\ 10000$, $w_i = 500$ and

$U = 1$. The verification finish time of three different algorithms is shown in **Fig. 4**, where SAVT_min is the test result of SAVT while the verifier is $VerS$ and $U = L$, and SAVT_max is the test result of SAVT while the verifier is $Ver1$ and $U = 1$. The verification finish time of EEAVT is very close to RRAVT in **Fig. 4(a)** and **Fig. 4(b)** whether each block length is 4KB or 8KB. Moreover, they are both far less than SAVT. Let $w_i = 500$ and $P = 90\%: 1\%: 100\%$, the verification finish time corresponding to the different detection rate is shown in **Fig. 4(c).** Also, both RRAVT and EEAVT have the same situation. In short, the verification finish time of both EEAVT and RRAVT is much shorter than SAVT under the different detection rate and the number of checked blocks. The reason is that multiple verifiers execute the verification in EEAVT and RRAVT.

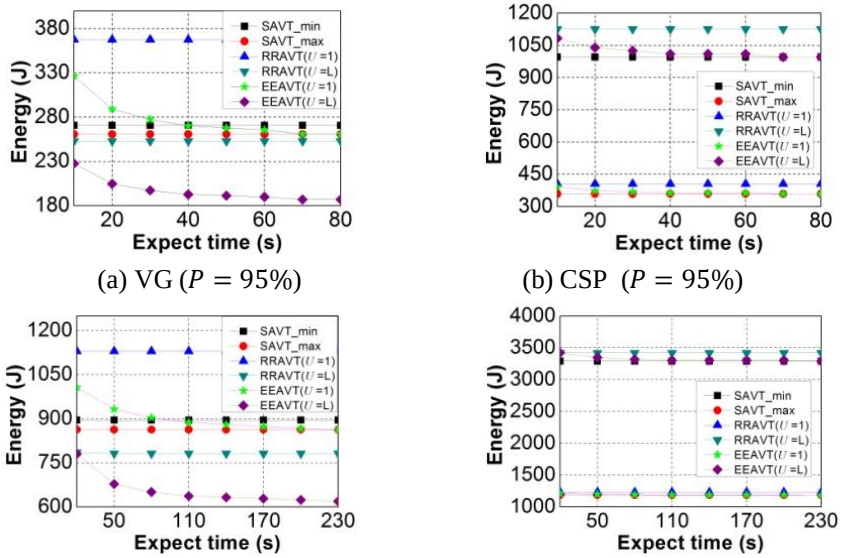


(a) number of blocks (4KB) (b) number of blocks (8KB) (c) detection rate

Fig. 4. The verification finish time

6.3 Verification Energy Cost

Let $T = 10: 10: 80$ ($P = 95\%$), and $T = 20: 30: 230$ ($P = 100\%$). The energy cost of VG and CSP is shown in **Fig. 5**. For different detection rate (i.e., $P = 95\%$ or $P = 100\%$) and response time (i.e., $U = 1$ and $U = L$), VG's and CSP's energy cost of the algorithm EEAVT is far less than RRAVT in the verification. Although the energy cost of SAVT is close to EEAVT, it has longer verification finish time and lower detection rate due to the assignment of its simple verification tasks.



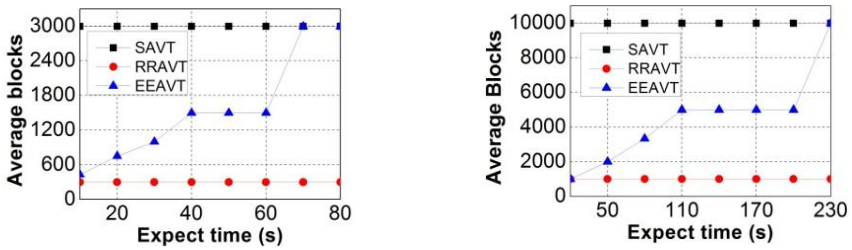
(a) VG ($P = 95\%$) (b) CSP ($P = 95\%$)

(c) VG ($P = 100\%$) (d) CSP ($P = 100\%$)

Fig. 5. The verification energy cost of VG and CSP

6.4 Verification Workload

The verification workload is the average number of blocks assigned to the verifiers. Let $U = 1$, $w_i = 500$ and the values of P are 95% and 100% respectively. The verification workload of three algorithms is shown in **Fig. 6**. RRAVT has the lowest verification workload for each verifier due to its average assignment of verification tasks. The verification workload of EEAVT is between RRAVT and SAVT since it chooses the verifier with low energy cost and dynamically assigns verification tasks according to the expected finish time.



(a) $P = 95\%$ (b) $P = 100\%$

Fig. 6. The verification workload

6.5 Experiment Analysis

By the previous tests, we can see that EEAVT has the advantage than RRAVT and SAVT in terms of verification finish time and verification energy cost. The reason is that EEAVT not only pays attention to the energy cost of verification tasks, but also constraints their verification finish time. Although SAVT limits the owner's finish time, the verifier must have enough operation time to finish the verification tasks. In this case, it is unsuitable for mobile devices since their limited energy and available operation time. Moreover, as both RRAVT and SAVT aren't aware of the energy cost of verification tasks, they cannot minimize the energy cost. On the other hand, EEAVT has the disadvantage of the verification workload than RRAVT since RRAVT aims in particular at balancing the load of verification tasks among multiple verifiers, and making full use of verifiers' operation time. Obviously, it is also unsuited to mobile devices. Although EEAVT doesn't balance the verification workload among multiple verifiers, the balance of the verification workload is insignificant since mobile devices have different energy reserve and available operation time. For mobile devices acting as the verifiers, it is more important to complete the verification tasks within the expected finish time. Thus, it should dynamically adjust the assignment of verification tasks with the change of verification request such as the expected finish time and the detection rate to effectively develop devices' perfromance other than exhaust their resources. Thus, EEAVT is more suitable to mobile devices and the scene of energy awareness.

7. Conclusion

Mobile devices which execute the data integrity verification have their inherent shortcomings, i.e., constrained resources. In this paper, we propose an energy-efficient algorithm for assigning verification tasks to get the credible verification results and minimize the energy cost of verification while the mobile devices act as the data verifiers. The algorithm utilizes multiple verifiers to improve the efficiency and reliability of the verification and satisfy owner's elastic verification request according to verifiers' residual energy and available

operation time. In the future, we further optimize the algorithm to balance the verification workload among multi verifiers.

References

[1] Mehdi Sookhak, Hamid Talebian, Ejaz Ahmed, Abdullah Gani, Muhammad Khurram Khan, "A review on remote data auditing in single cloud server: Taxonomy and open issues," *Journal of Network and Computer Applications*, Vol. 43, pp. 121-141, 2014. Article (CrossRef Link)

[2] Chang Liu, Chi Yang, Xuyun Zhang, Jinjun Chen, "External integrity verification for outsourced big data in cloud and IoT: A big picture," *Future Generation Computer Systems*, Vol.49, pp. 58-67, 2015. Article (CrossRef Link)

[3] Abdalla, Al-kindy Athman, Al-Sakib Khan Pathan, "On protecting data storage in mobile cloud computing paradigm," *IETE Technical Review*, Vol. 31, No.1, pp. 82-91, May 2014. Article (CrossRef Link)

[4] Jiawei Yuan, Shucheng Yu, "Public Integrity Auditing for Dynamic Data Sharing With Multiuser Modification," *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 8, pp. 1717-1726, 2015. Article (CrossRef Link)

[5] Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, Dawn Song, "Provable data possession at untrusted stores," in *Proc. of the 14th ACM conference on Computer and Communications Security*. ACM, pp.589-609, 2007. Article (CrossRef Link)

[6] Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, Jin Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, Vol.22, No.5, pp.847-859, 2011. Article (CrossRef Link)

[7] Yan Zhu, Gail-Joon Ahn, Hongxin Hu, S. S. Yau, H. G. An, Chang-Jun Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE Transactions on Services Computing*, Vol.6, No.2, pp.227-238, 2013. Article (CrossRef Link)

[8] Chang Liu, Jinjun Chen, Laurence T. Yang, Xuyun Zhang, Chi Yang, Rajiv Ranjan, Kotagiri Rao, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Transactions on Parallel and Distributed Systems*, Vol.25, No.9, pp. 2234-2244, 2014. Article (CrossRef Link)

[9] Shacham Hovav, Waters Brent, "Compact proofs of retrievability," in *Proc. of the 14th International Conference on the Theory and Application of Cryptology and Information Security*, pp.90-107, 2008. Article (CrossRef Link)

[10] Jia Yu, Kui Ren, Cong Wang, Vijay Varadharajan, "Enabling Cloud Storage Auditing With Key-Exposure Resistance," *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 6, pp. 1167-1179, 2015. Article (CrossRef Link)

[11] Huaqun Wang, "Identity-Based Distributed Provable Data Possession in Multicloud Storage," *IEEE Transactions on Sercices Computing*, Vol. 8, No. 2, pp. 328-340, 2015. Article (CrossRef Link)

[12] Ziad Ismail, Christophe Kiennert, Jean Leneutre, and Lin Chen, "Auditing a cloud provider's compliance with data backup requirements: a game theoretical analysis," IEEE *Transactions on Information Forensics and Security*, Vol.11, No.8, pp.1685-1699, 2016. Article (CrossRef Link)

[13] Jin Li, Xiao Tan, Xiaofeng Chen, Duncan S. Wong, Fatos Xhafa, "OPoR: Enabling Proof of Retrievability in Cloud Computing with Resource-Constrained Devices," *IEEE Transactions on Cloud Computing*, Vol. 3, No. 2, pp. 195-205, 2015. Article (CrossRef Link)

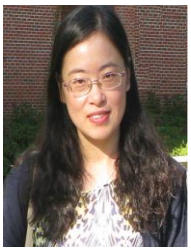
[14] Karthik Kumar, Yung-Hsiang Lu, "Cloud computing for mobile users: Can offloading computation save energy?," *IEEE Computer Society*, Vol.43, pp.51-56, 2010. Article (CrossRef Link)

[15] Shumao Ou, Kun Yang, Antonio Liotta, "An adaptive multi-constraint partitioning algorithm for offloading in pervasive systems," in *Proc. of IEEE International Conference on Pervasive Computing and Communications*, pp.116-125, 2006. Article (CrossRef Link)

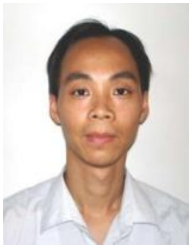
[16] Huijun Wu, Dijiang Huang, Samia Bouzefrane, "Making offloading decisions resistant to network unavailability for mobile cloud collaboration," in *Proc. of 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pp.168–177, 2013. Article (CrossRef Link)
[17] Yiyu Chen, Amitayu Das, Wubi Qin, Anand Sivasubramaniam, Qian Wang, Natarajan Gautam, "Managing server energy and operational costs in hosting centers," in *Proc. of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp.303-314, 2005. Article (CrossRef Link)
[18] Shang Li, Peh Li-Shiuan, Jha Niraj K., "Dynamic voltage scaling with links for power optimization of interconnection networks," in *Proc. of the Ninth IEEE International Symposium on the High-Performance Computer Architecture*, pp. 91-102, 2003. Article (CrossRef Link)
[19] Zhigang Deng, Zeng Guosun, and Wang Wei, "Energy Consumption Analysis Satisfying Time–Energy–Probability Constraints for Modern DVFS Microprocessor, " *IETE Technical Review*, Vol. 32, No 4, pp. 260-272, Feb. 2015. Article (CrossRef Link)
[20] Dzmitry Kliazovich, Pasca Bouvry, Samee Khan Ullah, "DENS: data center energy-efficient network-aware scheduling," *Cluster Computing*, Vol.16, No.1, pp. 65-75, 2011. Article (CrossRef Link)
[21] Dzmitry Kliazovich, Sisay T. Arzo, Fabrizio Granelli, Pascal Bouvry, Samee Ullah Khan, "e-STAB: energy-efficient scheduling for cloud computing applications with traffic load balancing," in *Proc. of the 2013 IEEE International Conference on Green Computing and Communications*, pp.7-13, 2013. Article (CrossRef Link)
[22] Yanzhi Wang, Xue Lin, Massoud Pedram, "A nested two stage game-based optimization framework in mobile cloud computing system," in *Proc. of The 7th IEEE International Symposium on Service Oriented System Engineering (SOSE)*, pp.494-502, 2013. Article (CrossRef Link)
[23] Weiwen Zhang, Yonggang Wen, Jianfei Cai, Dapeng Oliver Wu, "Toward transcoding as a service in a multimedia cloud: energy-efficient job-dispatching algorithm," *IEEE Transactions on Vehicular Technology*, Vol.63, No.5, pp. 2002-2012, 2014. Article (CrossRef Link)
[24] Weiwen Zhang, Yonggang Wen, Hsiao-Hwa Chen, "Toward transcoding as a service: energy-efficient offloading policy for green mobile cloud," *IEEE Network*, pp.67-73, Nov.2014. Article (CrossRef Link)
[25] Weiwen Zhang, Yonggang Wen, Kyle Guan, Dan Kilper, Haiyun Luo, Dapeng Oliver Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," I*EEE Transactions on Wireless Communications*, Vol.12, No.9, pp.4569-4581, 2013. Article (CrossRef Link)

Guangwei Xu is an associate professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. He received the M.S. degree from Nanjing University, Nanjing, China, in 2000, and the Ph.D. degree from Tongji University, Shanghai, China, in 2003. His research interests include the verification of data integrity, privacy protection, data security, QoS and routing of the wireless Ad Hoc networks and sensor networks.

Zhifeng Sun is a graduate in the School of Computer Science and Technology at Donghua University, Shanghai, China. His research interests include the verification of data integrity and data encryption.



Cairong Yan is an associate professor in the School of Computer Science and Technology at Donghua University, Shanghai, China. She received her Ph.D degree in computer science from Xian Jiaotong University of China in 2006. Her research interests include parallel and distributed computing, cloud computing, and big data processing.



Xiujin Shi is an associate professor in the school of computer science and technology at Donghua University, Shanghai, China. He received Ph.D. degree in computer science from Donghua University, China, in 2014. His research interests include big data processing and data security.

Yue Li is an associate professor in the school of computer science and technology at Donghua University, Shanghai, China. He received Ph.D. degree in electronic and computer engineering from University of Limerick, Ireland, in 2010. His research interests include computer network and security.