Telecommun Syst (2015) 60:303–313
DOI 10.1007/s11235-015-0031-8

Secured map reduce computing based on virtual machine using threshold secret sharing and group signature mechanisms in cloud computing environments

Hua Yi Lin[1] · **Meng-Yen Hsieh**[2] · **Kuan-Ching Li**[2]

Published online: 2 April 2015

Abstract Nowadays, cloud computing becomes a popular technology which combines resources of numerous physical computers and servers to perform distributed computing. The main benefit of cloud computing is that this technology decreases computing costs and infrastructures, allowing much more efficient computing. Through a portal, users submit working tasks and receive the results without assigning to specific servers. Nevertheless, the computers of users and enterprises are located in cloud, and arbitrary clients can randomly log on and take private data away. Thus, the cloud security becomes a significant subject. In this paper, we exploit the threshold crypto sharing (Desmedt and Frankel in Advances in cryptology—CRYPTO'89, 1990) with group signature mechanism to secure transmitted data. During map and reduce phases, this mechanism can protect the divided and merged messages from being tampered with. Additionally, this study exploits a virtual machine platform to simulate cloud computing environments and then perform security operations. Experimental results show that the mechanism presented has lower cost comparing to other existing ones and very promising its application in cloud environments.

✉ Hua Yi Lin
calvan.lin@msa.hinet.net

Meng-Yen Hsieh
mengyen@pu.edu.tw

Kuan-Ching Li
kuancli@pu.edu.tw

[1] Department of Information Management, China University of Technology, Taipei, Taiwan, ROC

[2] Department of Computer Science and Information Engineering, Providence University, Taichung, Taiwan, ROC

Keywords Cloud security · Threshold crypto sharing · Group signature · Map · Reduce · Virtual machine

1 Introduction

Map/Reduce mechanisms in cloud computing are proposed by Google, which are based on hundreds or thousands of unreliable computers and servers providing computing cycles concurrently, and comprises three kinds of roles, including Mapper, Reducer and Master computers [1]. The role of Master computer takes charge of acquiring demand duties from clients, separating duties into several operations, and appointing operations to the related Map and Reduce computers. As a Master computer accepts Map/Reduce duties from remote users, then assigns Mapper and Reducer computers to cooperatively execute Map/Reduce operations. Figure 1 depicts the map and reduce operation and framework [2,3]. Additionally, programmers need to specify the parallel procedure, and write Map/Reduce applications to execute computing on several computers in the cloud framework. Finally, every Reducer gathers the temporary data coming from Mappers, and then assembles the piece of data into an entire data [4].

Since the Master computer takes charge of deciding which members to execute the Map and Reduce functions, and assigning the Map duty to the Mapper computers. Subsequently, the Master computer separates the received data into numerous datagram. At the end of computing, each Mapper stores the intermediate data into its storage. Subsequently, Reducer computers obtain the intermediate data from Mapper computer, and execute reduce operations to combine intermediate data into an entire data.

However, the entire infrastructure lacks secure operations to protect the transmitting data among the Master,

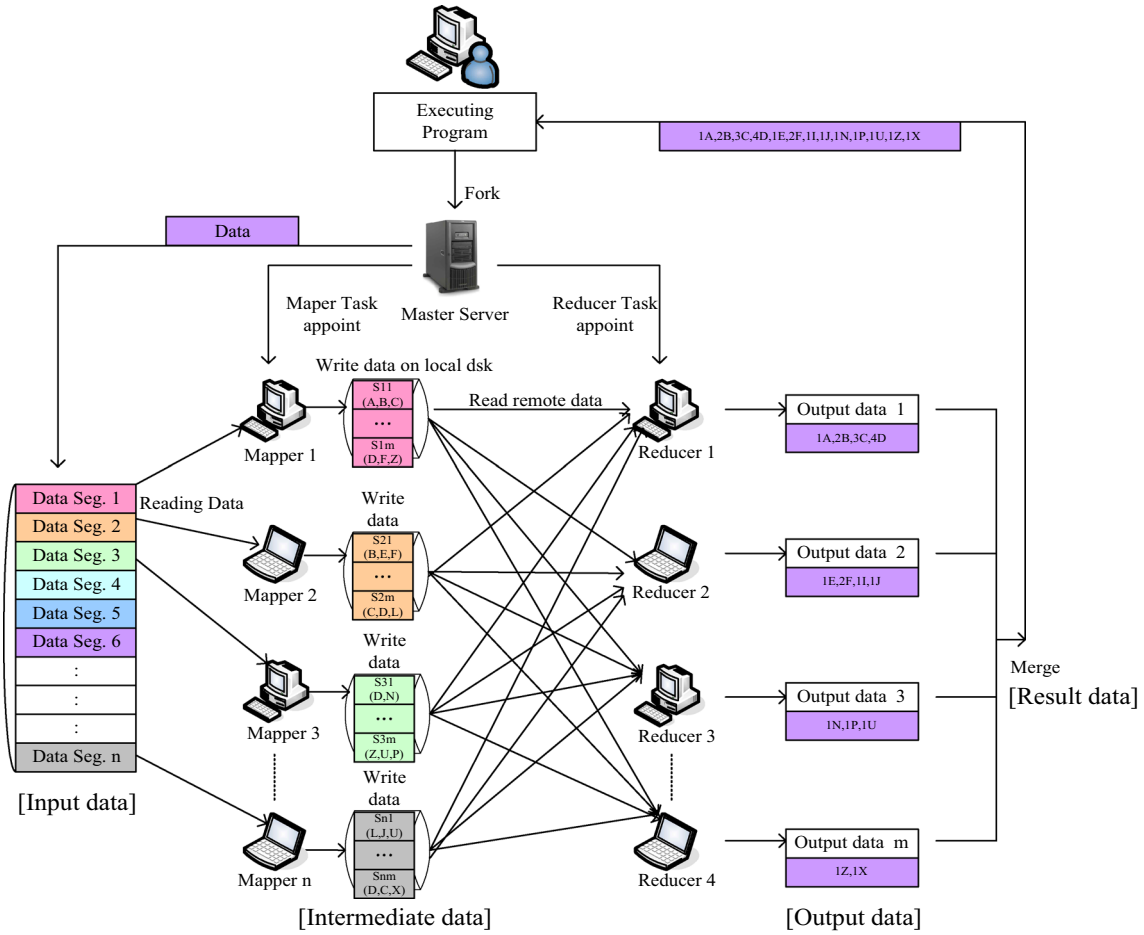

Fig. 1 The operation stages of Map and Reduce

Mappers and Reducers. Therefore, the infrastructure cannot ensure data confidentiality. Besides, the entire infrastructure lacks authentication mechanism to ensure the identification of each other. A malicious computer probably impersonates a Mapper (Reducer) or declares that itself is a Master (Mapper/Reducer). Subsequently, it initiates a Map/Reduce operation to perform spoofing attacks. Moreover, during data transmissions, there are no mechanisms to ensure the data integrity. Mappers and Reducers probably modify or send incorrect messages to each other. Eventually, users obtain an incorrect result [5,24].

Another important issue, the operation in cloud is distributed across the Internet and difficult to implement related experiments. Thus, we construct a virtual machine architecture to simulate the cloud environment and execute map/reduce operations. Additionally, this study exploits threshold secret sharing and group signature schemes to secure data transmissions and achieve secure Map Reduce operations. This paper focuses on how to strengthen secure Map/Reduce functions, and exploits the threshold crypto sharing with group signature schemes to achieve the above purpose. Additionally, the hash message authentication code (HMAC) is an efficient mechanism to validate the data integrity during transmission. Therefore, we develop several mechanisms to enhance secured data transmissions and ensure the identification of participators.

The organization of this paper is constructed as follows. Section 2 presents the mechanism of threshold crypto sharing. The secure group signature scheme is depicted in Sect. 3. The mechanism of securing Map/Reduce data transmission is proposed in Sect. 4. Analyses of the computing evaluation and simulation results are described in Sect. 5. Section 6 draws the conclusions and future work.

2 Mechanism of the threshold crypto sharing

Nowadays, in cloud computing, majority of operational environments are constructed on virtual machine frameworks to execute concurrent computing, as depicted in Fig. 2. In our research, we specific that a virtual machine (VM) titled PKI VM performs the Master role, and numerous VMs execute

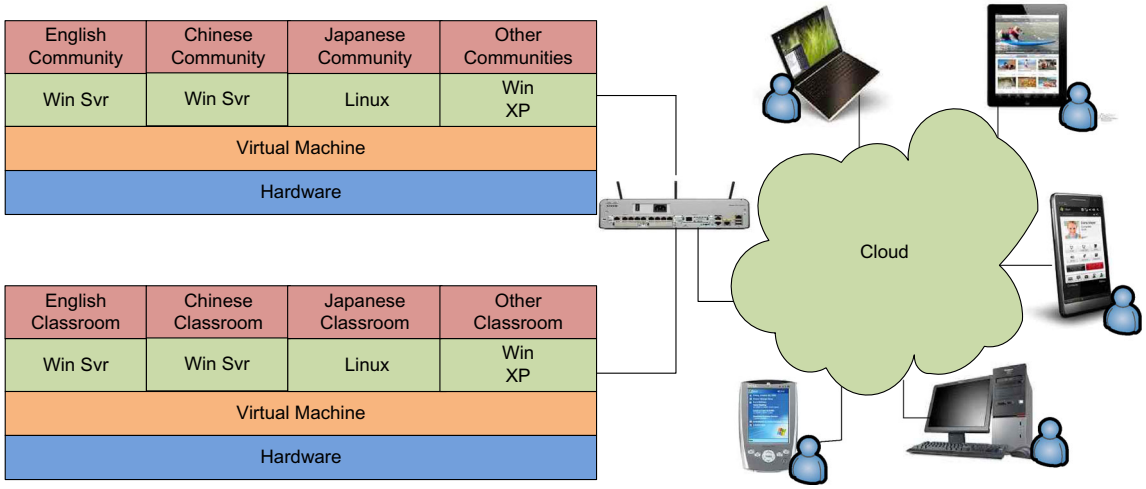

Fig. 2 The framework of virtual machine in cloud environment

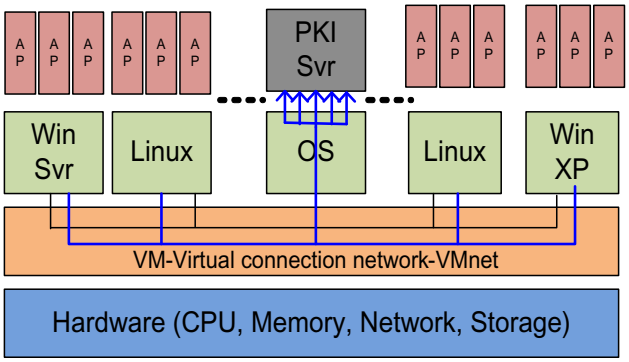

Fig. 3 PKI in virtual machine framework

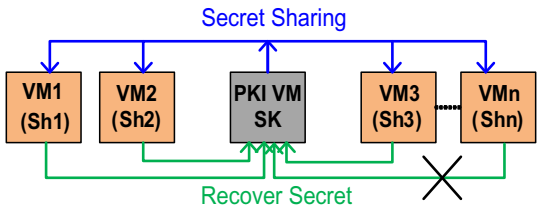

Fig. 4 A $(n, 3)$ threshold crypto secret sharing recomputes the system-secret-key SK

the role of Reducers, while some VMs play the role of Mappers. Figure 3 depicts the public key infrastructure (PKI) in virtual machine framework. For the purpose of achieving secure cloud computing, this study enhances the threshold crypto sharing mechanism and exploits the group signature scheme to protect the data transmission. The proposed scheme concentrates on strengthening the secure capability and reducing the vulnerability of the cloud computing. In this study, we abandon the centralization authority server, and spread the operations of authorization among numerous VMs. Thus, this mechanism can achieve fault tolerance and avoid a single point of failure. In the early stage, the similar ideas have been presented in different fields, but this mechanism is well suited to be adopted on cloud computing environments.

In cloud computing, this study would like to equip with perfect secret and fault tolerance abilities. Therefore, we determine to adopt a threshold crypto scheme [6,7]. In a (n, m) threshold crypto scheme, it permits n Mapper VMs collectively to perform a crypto operation and generate a secret. In case, the secret is damaged. Any m Mapper VMs collectively execute the recovery operations, and then obtain the original secret. However, a maximum of $m - 1$ Mapper VMs cannot recover the secret. In another word, the system-secret-key SK is the secret, andSK can be recovered by m Mapper VMs.

This study would like to assure that the operations can endure $m - 1$ compromised Mapper VMs. Therefore, we adopt a $(n,$ m) threshold crypto scheme, and separate SK into n shadows, such as $Sh_1, Sh_2, \ldots, Sh_n$, named crypto sharing keys, and finally every Mapper VM_i owns a shadow Sh_i. For example, a $(n, 3)$ threshold crypto scheme is depicted in Fig. 4. The accurate Mapper VMs (1, 3 and n) deliver their own crypto sharing key Sh_i to the dealer PKI VM. In case, the other Mappers $VM_i (i \neq 1, 3$ and $n)$ decline to deliver their own crypto sharing key. The PKI VM instantly recomputes the system-secret-key SK from Mappers VM_1, VM_3 and VM_n. But, the maximum of two compromised Mapper VM_i cannot compute SK, even if they conspire, since they only have two partial crypto sharing key.

Afterward, this study exploits the crypto sharing scheme on secure Map/Reduce functions. At the beginning, the PKI VM computes the system-secret-key SK. Subsequently, PKI VM separates SK into n shadows, such as $Sh_1, Sh_2, \ldots, Sh_n$. Initially, the PKI VM takes charge of assigning n shadows

to n Mapper VMs. At the end of the assigning operations, each Mapper VM_i has a shadow Sh_i. Even if the whole system breaks down, the substitute PKI VM can recompute the original system-secret-key SK via gathering $k-1$ shadows from the Mapper VM_i.

In our proposed crypto sharing mechanism, we denote the system key pair as (PK, SK), which is generated by the PKI VM. Moreover, SK represents the system-secret-key, and PK indicates the system-public-key. When a computer departs or joins the system, which needs to recompute a new system key pair (PK, SK) via PKI VM using the aforementioned mechanism. After regenerating (PK, SK), each Mapper VM_i receives the renewed PK, and the system separates SK into n shadows, such as $Sh_1, Sh_2, \ldots, Sh_n$, via the crypto sharing mechanism. The PKI VM unicasts Sh_i to the Mapper VM_i using a secure transmission way. After receiving, the Mapper VM_i owns a key pair (Pch_i, Sh_i) for further communications, and SK is shared by random Mapper VMs via a crypto polynomial $f(x)$. In case, the polynomial $f(x)$'s degree is $m-1$, then any m Mapper VMs has the ability to regenerate the system-secret-key SK via Lagrange Interpolation. However, the vicious Mapped VMs likely disclose SK, when any numbers of Mapper VMs is not more than m, they can reconstruct SK. At the beginning, the PKI VM calculates the system-secret-key $SK = S_d$, and arbitrarily chooses a polynomial $f(x)$ with degree $m-1$, $f(x) = S_d + f_1 x + \cdots + f_{k-1}x^{m-1}$, where S_d and $f_i (i = 1, 2, \ldots, m-1)$ are not more than arbitrary prime p. Besides, $f(0) = S_d \bmod p$ indicates the system-secret-key. As this study mentioned above, every Mapper $VM_i (i = 1, 2, \ldots, n)$ owns a shadow $Sh_i = (f(VM_i) \bmod p)$. When m Mapper VMs $(VM_1, VM_2, VM_3 \ldots, VM_m)$ would like to regenerate the system-secret-key SK, we can infer it from the following equation

$$S_d \equiv \left[\sum_{i=1}^{m} Sh_i \bullet l_{\mathrm{VM}i}(0)\right] \bmod p \equiv \left[\sum_{i=1}^{m} SK_i\right] \bmod p,$$

where the Lagrange Coefficient is $l_{VMi}(x) = \prod_{j=1, j\neq i}^{m} \frac{(x-VM_j)}{(VM_i-VM_j)}$. By Lagrange Interpolation, every share owner VM_i has the ability to compute SK_i via its own Sh_i, and the $S_d (= SK)$ can be regenerated from the following equation

$$S_d = \left[\sum_{i=1}^{m} SK_i\right] \bmod p.$$

Since enemies probably intrude into m or more Mapper VMs in sufficient time. In order to protect the crypto sharing system from attacks. The crypto sharing system needs to be renewed periodically with various polynomials via a predictive approach. Therefore, we infer a new polynomial f_{m+1} from f_m, $f_{m+1} = f_m + g_m$, where g_m is an arbitrary polynomial with degree $(m-1)$, and the renew crypto sharing $f_{m+1}(Mp_i) = f_m(VM_i) + g_m(VM_i)$ has the ability to recompute the system-secret-key S_d. In case, a Mapper VM_i's shadow is disclosed or expired, then the Mapper VM_i is regarded as an insecure member. Thus, Master saves the identification of the Mapper VM_i and its shadow key into the certificate revocation list (CRL) by an insecure factor, and then delivers this information to the other Mapper VMs. After receiving the information, the Mapper VMs store the identification of the charged Mapper VM_i into their revocation list and reduce its secure ranking. When a Mapper VM_i obtains m charges from other Mapper VMs convicting of its insecure actions, then the Mapper VM_i will be refused to cooperate with. According to the above mechanism, the PKI VM has the ability of fault tolerance. When the PKI VM breaks down or any m Mapper VMs accuse it of insecure actions, this system selects another PKI VM among VMs. Subsequently, the PKI VM collectively recomputes SK through $m$shadows $Sh_1, Sh_2, \ldots, Sh_n$ in the rest of Mapper VMs, and then the system operates regularly.

In cloud computing environments, since the system cannot ensure which Mapper/Redurer VM_i will join or leave the operations, and therefore our scheme provides a dynamic mechanism to reconstruct the SK efficiently. Subsequently, we integrate the secret sharing key into secure data transmissions to perform Map/Reduce functions. During the data transmission, this study exploits the receiver's shadow Sh_i to sign and cipher the delivered data. Subsequently, the sender delivers the results to the receiver. Once the Mapper VM_i receives the delivered data, VM_i verifies the accuracy of the secret sharing key Sh_i and decrypts the encrypted data. Additionally, the cloud computing environments include many participants. In order to ensure the identification of each other and protect the transmitted data, this study adopts group signature to ensure accuracy of the collective computing data. The detailed description is presented as follows.

3 Secure group signature

3.1 Trusted (n, m) threshold group signature scheme

This section introduces a group verification mechanism to enhance the security of cloud computing environments. We exploit Shamir's (n, m) threshold scheme and specific module operations [8,25]. Meanwhile, the PKI VM plays the role of trusted key center (TKC) also is responsible for determining common parameters, including all private keys for member VMs. The related parameters are as follows:

This system selects a big prime number p between 2^{511} and 2^{512}. Additionally, $p-1$ is divisible by q which is a prime between 2^{159} and 2^{160}. Subsequently, we choose the

coefficient $\{a_x, x = 0, 1, 2, \ldots n-1\}$ of the polynomial $f(x) = a_0 + a_1 + \cdots + a_{t-1}x^{n-1} \bmod q, a_x \in [1, q-1]$. Besides, $g = n^{p-1/q} \bmod p > 1$, where n is a random number and g is a generator with series q in $GF(p)$, and $\{p, q, g\}$ is public and $\{a_x\}$ is private.

Subsequently, the PKI VM chooses the $y = g^{f(0)} \bmod p$ as a public key for the group, and chooses a public value x_i and the $f(x_i) \bmod q$ as a private key for each VMi. Eventually, the PKI VM calculates $y_i = g^{f(xi)} \bmod p$ for each VM_i as the public key.

Mapper VMs $(m_1, m_2, m_3, \ldots, m_n)$ have to represent the group of VMs $(m_1, m_2, m_3, \ldots, m_m)$ to sign the *Data*. As each m_x receives a partial message $(Data_{Segi})$ from the PKI VM, $m_x[1 \leqq x \leqq n]$ signs the $Data_{Segi}$. Then m_x chooses an integer $t_i \in [1, q-1]$, calculates $w_i = g^{ti} \bmod p$ as the commitment, and broadcasts $\{w_i\}(i = 1, 2, 3, \ldots, n)$ to the other $m_y[y \neq x]$. After broadcasting, each Mapper VM calculates its

$$w = \prod_{i=1}^{n} w_i \bmod p, \tag{1}$$

and uses its private key $f(x_i) \bmod q$ and t_i to sign $Data_{Segi}$.

$$Data_{Signi} = f(x_i) Data_{Seg}{}' \left(\prod_{j=1, j\neq i}^{n} \frac{-x_j}{x_i - x_j} \right) - t_i w \bmod q \tag{2}$$

Here, $\{w_i, Data_{signi}\}$ is a personal signature, and satisfies $Data_{Segi}{}' = f(Data_{Segi})$. When $\{w_i, Data_{Signi}\}$ is delivered to the PKI VM (TKC), the PKI VM uses each VM_i's public value x_i and public key y_i according to the following equation to verify the personal signature $\{w_i, Data_{Signi}\}$:

$$y_i^{Data_{Segi}{}'} \left(\prod_{j=1, j\neq i}^{n} \frac{-x_j}{x_i - x_j} \right) = w_i^w g^{Data_{Sign_i}} \bmod p \tag{3}$$

As the PKI VM receives each $Data_{Signi}$ and verifies n signatures of VMs, then merges each VM_i's signature, w_i and $Data_{signi}(i = 1, 2, 3, \ldots, n)$ into a group signature $\{w, Data_{Sign}\}$, where $Data_{Sign} = Data_{Sign1} + Data_{Sign2} + Data_{Sign3} + \cdots + Data_{Signn} \bmod q$. Therefore, if any unauthorized VM join the Map/Reduce operations, the group signature results will be incorrect. Therefore, this system can verify whether the reduced data is modified during the transmission.

3.2 Verification of the (n, m) group signature

This section presents how to verify the correctness of a group signature in the VM cloud computing environment. Initially, the PKI VM announces a public key y for the group member VMs, which is used to verify the group signature $\{w, Data_{Sign}\}$ of the transmitted data. The verification equation is $y^{Data_{Segi}{}'} = w^w g^{Data_{Sign}} \bmod p$. In case that the above equation is true, and the group signature $\{w, Data_{Sign}\}$ can be correct.

Inference

Since each VM_i's signature $\{w_i, Data_{Signi}\}$ satisfies the following equation.

$$y_i^{Data_{Segi}{}'} \left(\prod_{j=1, j\neq i}^{n} \frac{-x_j}{x_i - x_j} \right) = w_i^w g^{Data_{Sign_i}} \bmod p. \tag{4}$$

According to the Eq. (4), if we multiply the above equation n times $(i = 1, 2, 3, \ldots n)$, then replace $y_i^{Data_{Segi}{}'} \left(\prod_{j=1, j\neq i}^{n} \frac{-x_j}{x_i - x_j} \right)$ with $w_i^w g^{Data_{Sign_i}} \bmod p$. We can infer the following equations.

$$\prod_{j=1}^{n} y_i^{Data_{Segi}{}'} \left(\prod_{j=1, j\neq i}^{n} \frac{-x_j}{x_i - x_j} \right) = \prod_{i=1, j\neq 1}^{n} w_i^w g^{Data_{Signi}} \bmod p. \tag{5}$$

Subsequently, this study uses Eqs. (1) and (2) to verify whether $y^{Data_{Segi}{}'}$ is equivalent to $w^w g^{Data_{Sign}} \bmod p$. Eventually, the Eq. (8) indeed satisfies the results.

$$g^{Data_{Segi}{}'} \sum_{i=1}^{i} f(x_i) \prod_{j=1, j\neq i}^{n} \frac{-x_i}{x_i - x_j} \bmod p = \left(\left(\prod_{i=1}^{n} w_i \right)^{wg \sum_{i=1}^{n} Data_{Signi}} \bmod p \right) \tag{6}$$

$$g^{Data_{Segi}{}' f(0)} = w^w g^{Data_{Sign}} \bmod p \tag{7}$$

$$y^{Data_{Segi}{}'} = w^w g^{Data_{Sign}} \bmod p \tag{8}$$

From the above inferences, we can verify the accuracy of the group signature, and prevent the impersonation, modification, DoS, and repudiation flaws (among many others).

4 The secure data transmission mechanism on map/reduce

During the operation of Map/Reduce, the system likely runs into vicious assaults [11,12]. Therefore, the system needs to secure the whole operations among Mapper VMs, Reducer VMs and a PKI VM. Additionally, the system has to assure

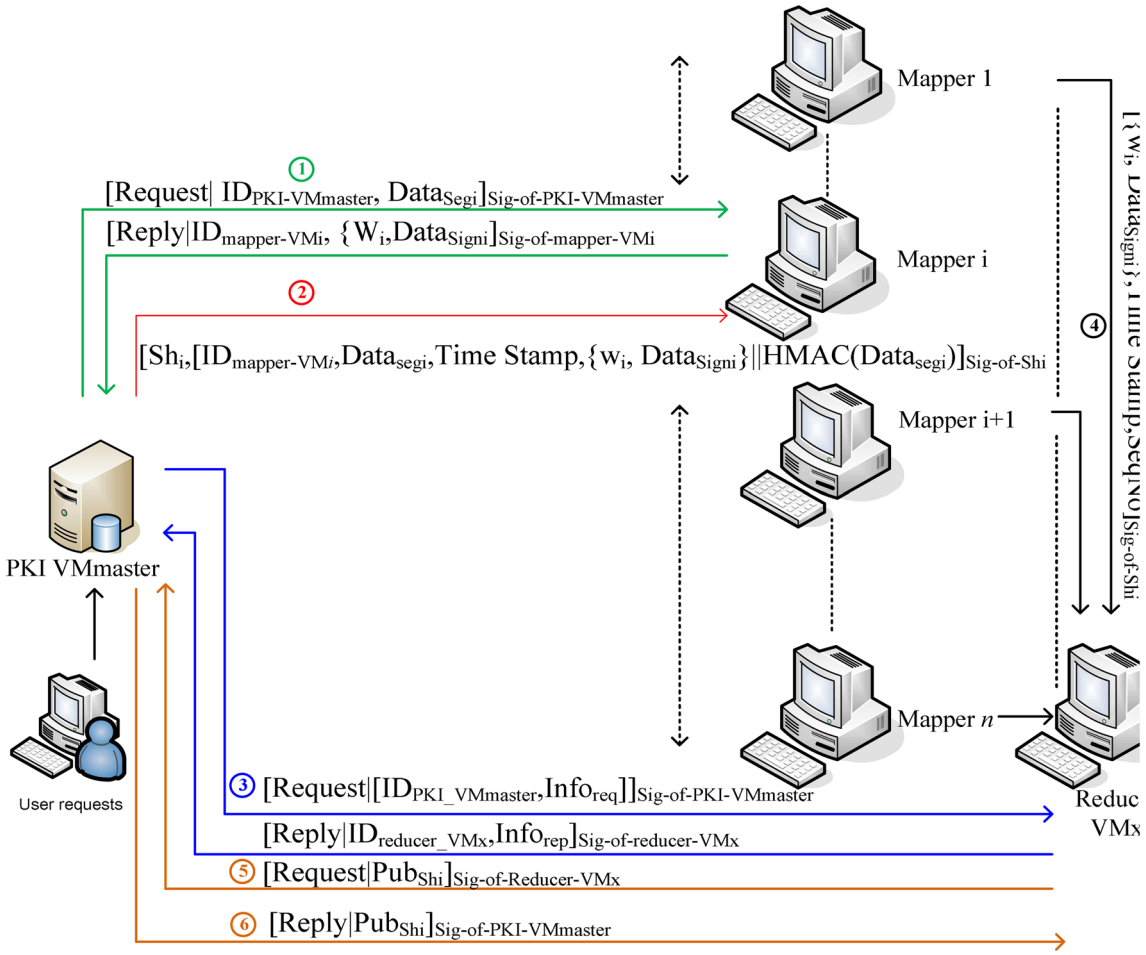

Fig. 5 The detailed procedure of the identification verification and secured Mapping/Reducing stages

the participating Mapper VMs are totally secure, and keep vicious nodes from imitating the Mapper VMs. Besides, the Reducer VMs rebuild the whole data via the intermediate data coming from Mapper VMs. The Reducer VMs have to assure the integrity of received data and correct identification of Mapper VMs to evade obtaining tampered message. Therefore, this study uses the threshold crypto sharing mechanism and group signature mechanism to protect Map/Reduce functions. Figure 5 depicts the secure Map/Reduce operations, and the following description presents the detailed processes.

4.1 Mapping stage

Stage 1: Initially, as the PKI VM_{master} obtains a user's demanding, it determines which Mappers will take part in the operations, and then sends the appointed duty to Mapper VM_i. In order to assure the correctness of the identity of the demander PKI VM_{master}, the PKI VM_{master} must sign the request, the identification $ID_{PKI-VMmaster}$ of PKI VM_i and data segment $Datas_{Segi}$, for subsequent verification on the identification of the PKI VM_{master}. Subsequently, the PKI VM_{master} delivers the entire result to the Mapper VM_i.

$$[Req|ID_{PKI-VMmaster}, Data_{Segi}]_{Sig-of-PKI-VMmaster}$$

Once obtaining the requisition, the Mapper VM_i uses the PKI VM_{master}'s public key to validate the correctness of the identification of the PKI VM_{master}, and signs the confirmation message of reply, the identification $ID_{mapper-VMi}$ of Mapper VM_i and the data segment $\{w_i, Data_{Signi}\}$.

$$[Rep|ID_{mapper-VMi}, \{w_i, Data_{Signi}\}]_{Sig-of-mapper-VMi}$$

Stage 2: Afterward, the PKI VM_{master} uses a hash function to compute the HMAC($Data_{segi}$) of the data segment $Data_{Segi}$, and then accumulates the identification $ID_{mapper-VMi}$ of Mapper VM_i, orignal data segment $Data_{segi}$, $Timestamp$, and partial group signature results as $\{w_i, Data_{Signi}\}$. Finally, the PKI VM_{master} uses the receiver's crypto sharing key Sh_i to sign the whole data. Subsequently the PKI VM_{master} delivers the entire signa-

ture data to the Mapper VM_i.

$$[Sh_i, [ID_{mapper-VMi}, Data_{segi}, TimeStamp, \{w_i, Data_{Signi}\} \parallel HMAC(Data_{segi})]]_{Sig-of-Shi}$$

As the Mapper VM_i obtains the delivered message, the Mapper VM_i deciphers the ciphered message, validates the integrity of HMAC($Data_{segi}$), and determines the correctness of the crypto sharing key Sh_i.

4.2 Reducing stage

Stage 3: When a Reducer VM_x obtains an appointed duty coming from the PKI VM_{master}, and to be asked executing the reduce process. In order to assure the correctness of the demander, the PKI VM_{master} has to sign the request data, the identification $ID_{PKI-VMmaster}$ of PKI VM_{master}, and the delivered information $Info_{req}$ for subsequent verification on the PKI VM_{master}'s identification.

$$[Req|[ID_{PKI-VMmaster}, Info_{req}]]_{Sig-of-PKI-VMmaster}$$

Once a Reducer VM_x obtains the delivered message, the Reducer VM_x validates the PKI VM_{master}'s identification via the PKI VM_{master}'s public key. Subsequently, the Reducer VM_x signs the reply data, the identification $ID_{reducer}$ of the Reducer VM_x, and the delivered message.

$$[Rep|ID_{reducer-VMx}, Info_{rep}]_{Sig-of-reducer-VMx}$$

Stage 4: Continually, a Reducer VM_x obtains the delivered data segment with signature $\{w_i, Data_{Signi(i=1\sim n)}\}$, $Time\ Stamp_{i(i=1\sim n)}$ and sequential number $SeqNo_{i(i=1\sim n)}$ [9,10] from the Mappers $VM_{(1\sim n)}$, which are ciphered by the Mapper VM_i's crypto sharing key Sh_i.

$$\text{Mapper } VM_{(1\sim n)} \rightarrow \text{Reducer} VM_x$$
$$[\{w_i, Data_{Signi(i=1\sim n)}\}, TimeStamp_{i(i=1\sim n)}, SeqNo_{i(i=1\sim n)}]_{Sig-of-Shi}$$

Stage 5: Once the Reducer VM_x obtains the delivered data segment from the Mappers $VM_{(1\sim n)}$, the Reducer VM_x demands the Mapper VM_i's corresponding public key of Sh_i from the PKI VM_{master} for later deciphering the delivered data.

$$\text{Reducer } VM_x \rightarrow \text{ PKI } VM_{master}$$
$$[Req|Pub_{Shi}]_{Sig-of-Reducer-VMx}$$

Stage 6: A Reducer VM_x obtains the Mapper $VM_{(1\sim n)}{}'s$ corresponding public key of Sh_i from the PKI VM_{master}.

$$\text{PKI } VM_{master} \rightarrow \text{ Rudcer } VM_x$$
$$[Rep|Pub_{Shi}]_{Sig-of-PKI-VMmaster}$$

Finally, a Reducer VM_x acquires the Sh_i's corresponding public key. Subsequently it deciphers the ciphered data, and combines every VM_i's signature $\{w_i, Data_{Signi(i=1\sim n)}\}$ into a group signature $\{w, Data_{Sign}\}$, where $Data_{Sign} = Data_{Sign1} + Data_{Sign2} + Data_{Sign3} + \ldots + Data_{Signn}$ mod$q$. Then, the Reducer VM_x validates the group signature $\{w, Data_{Sign}\}$ using the public key y. Afterward, the Reducer VM_x combines every portion of the delivered datagram into an entire data. As a result, the Reducer VM_x delivers the entire message to the demander, and thus completes the identification verification and secured Mapping/Reducing stages.

5 Analyses of security and computing evaluation

This section describes the analysis of security issues on the proposed mechanism, and simulates several scenarios to evaluate the efficiency of the proposed scheme.

(1) Threshold crypto sharing mechanism on fault tolerance:

In case, the PKI VM_{master} breaks down, in terms of the threshold crypto sharing mechanism with threshold values (n, m), this system gathers the maximum of m crypto sharing keys from VMs. Subsequently, this system uses the Lagrange Interpolation to recompute the system-secret-key SK. As a result, the whole system recovers instantly. Moreover, this study uses threshold group signatures to validate the correctness of the delivered data. Even numerous VMs confront vicious attacks, at least m VMs must cooperatively sign the data for the whole group, thus the system can conclude if the delivered data are tampered with.

(2) Confidentiality and authentication:

This study exploits signature to verify Master, Mapper and Reducer identifications. Only the participators with accurate signatures can pass through the authentication. Additionally, Reducer VMs and Mapper VMs exploit a signature and a public key to cipher the delivered data. Merely the VMs have the corresponding private key can decipher the ciphered message. The rest of computers don't realize the private key, and thus they cannot decipher the ciphered message. Therefore, the proposed mechanism ensures the confidentiality.

(3) Data accuracy and integrity:

After receiving data from Mapper VMs, the Reducer VM verifies whether the data are modified using a threshold group

signature. Additionally, this study calculates HMAC for each $Data_{segi}$ to assure the integrity and correctness of the delivered data. Through the data transfer, the sender uses a hash function and its private key to calculate HMAC. When the receiver acquires the delivered data, it uses its private key and the original data as inputs, and recomputes HMAC to validate the integrity. Because the hash function is irreversible, given an arbitrary number m, there are no methods to figure out n and make $H(n) = m$. Additionally, when m is not equal to n, $H(m)$ will not equal $H(n)$. Thus, if the recipient discovers the unmatched HMAC code, the recipient can assure that vicious computers tamper with the delivered message during the data transfer.

(4) Ciphering and deciphering operations:

In cloud computing environment, the remote computer's resources and computing power are imperceptible. In our proposed framework, only the PKI VM needs a powerful computing capability. Reducer and Mapped VMs just store a key pair to execute cipher and decipher operations, and conserve a HMAC-160 hash function or a RIPEMD-160 hash function to validate delivered data integrity. Therefore, the whole system only wastes a few energies. As a result, the proposed scheme is appropriate for cloud environments.

(5) Impersonation attacks and deny of service:

To prevent the impersonation attacks, this study adopts the personal signature and group signature scheme to ensure the identity of each other. Since each operation has a signature from the executive. Thus, the executive cannot repudiate its previous actions later. Additionally, The Reducer and Mapped VMs can employee their crypto sharing key to validate the received data integrity, a sequential number and the time stamp, and thus the proposed mechanism keeps the system from vicious reply assaults. Additionally, at the appointing job stages, the PKI VM uses a signature to assure the identification of the Reducers VM_i and Mapper VM_i. Therefore, this mechanism evades deny of service or impersonation assaults.

(6) Performance evaluation:

Recently, several studies improve the security of cloud computing environments using Kerberos. This study compares our proposed scheme with Kerberos. Since Kerberos needs an authentication server (AS) and a ticket granting server (TGS) [22,23], the entire security operation focuses on performing authentication for each other and creates the secret key for communications. Therefore, the secure data transmission is additional operations using Kerberos in cloud computing environments. We evaluates the response time, which a user sends a request till he receives the reply message under deferent scenarios. Besides, this study estimates the influence of increasing the number of Mappers and Reducers, and variable threshold values. Additionally, this study compares our proposed schemes with Kerberos on the secure data transmission time, and evaluates the calculating time for SK and Sh_i in the threshold scheme versus the application time for tickets and secret keys in Kerberos.

In Fig. 6, as we increase the message length with group signature schemes, the operational time of Map/Reduce raises stably, and the response time of our scheme outperforms Kerberos. In Fig. 7, although this study raises the number of duties, the response time of Map/Reduce raises desirably. Under fixed data length, Figs. 8 and 9 show that increasing the number of Mappers and Reducers, the elapsed time of performing secure operations still keeps stable growth [13,14]. Figure 10 indicates that this study adopts a $(n, 15)$ threshold crypto mechanism to permit n Mapper VMs to cooperatively execute a crypto function. We increase the threshold value gradually, and evaluate the elapsed time of receiving a result. Initially, the elapsed time increases abruptly. Over the fixed threshold value, the elapsed time increases stably. Figure 11 demonstrates the elapsed time [15–17] for reconstructing the system key SK on the variable threshold value, while

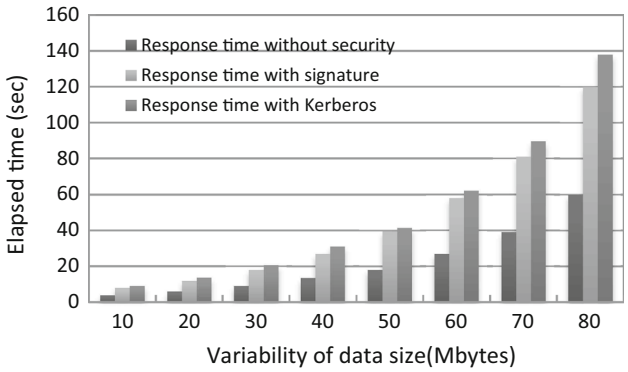

Fig. 6 The response time on different message length

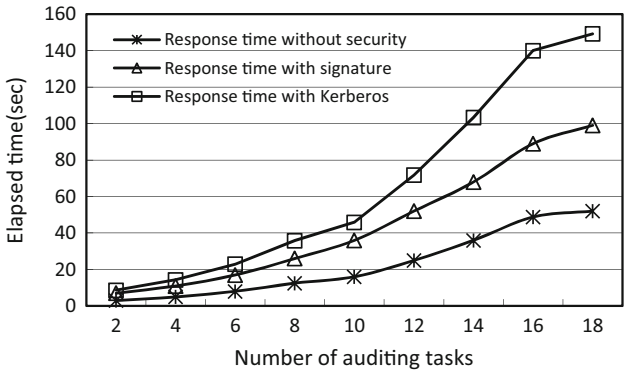

Fig. 7 The response time for raising tasks on different scenarios

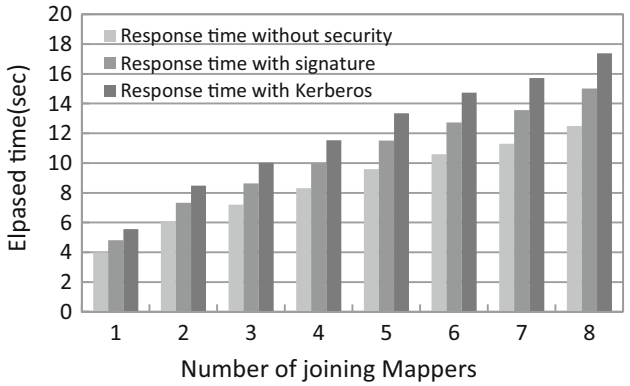

Fig. 8 The comparison of response time for increasing Mappers

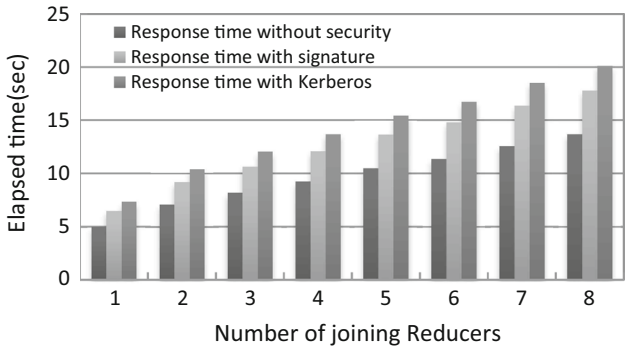

Fig. 9 The comparison of response time for increasing Reducers

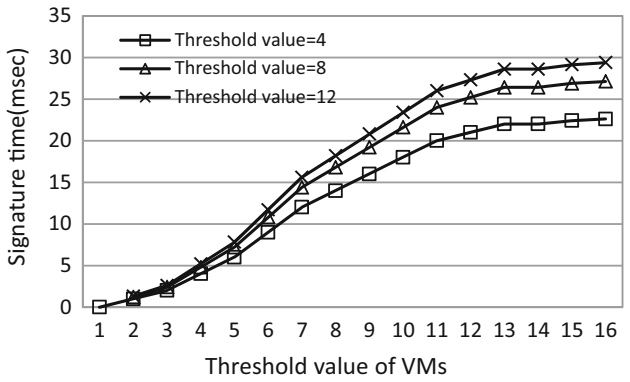

Fig. 10 The group signature time for variable threshold values

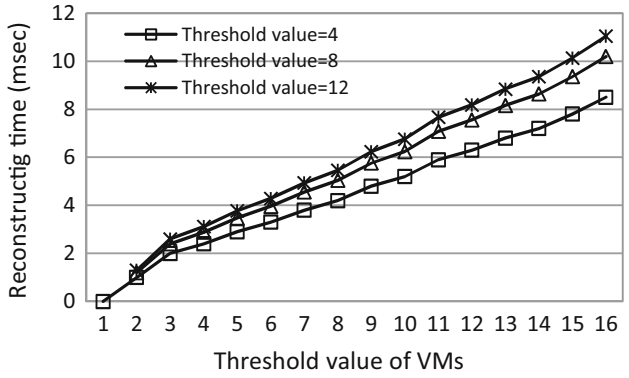

Fig. 11 The reconstructing time for system secret key *SK*

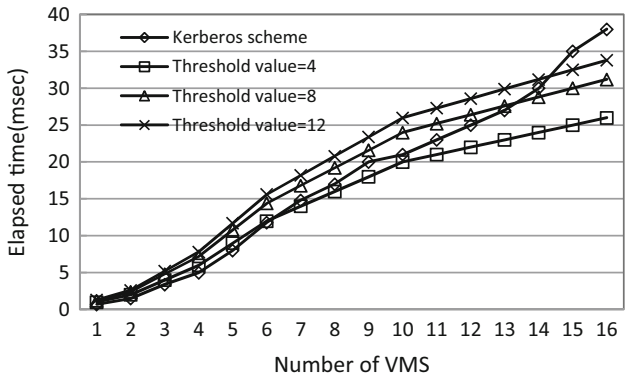

Fig. 12 The calculating time for *SK* and Sh_i in the threshold scheme versus the application time for tickets and secret keys in Kerberos

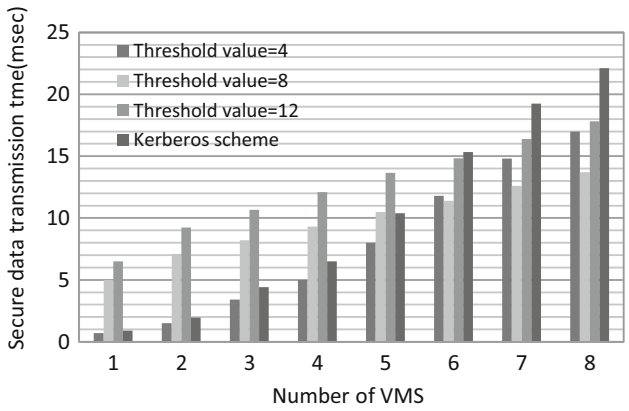

Fig. 13 The secure data transmission time for the threshold scheme versus Kerberos

increasing the threshold value, the system increases stably to rebuild the *SK*. Figure 12 demonstrates that our proposed schemes need at initial stage to calculate the crypto sharing key Sh_i and *SK* for Mappers/Reducers, and hence consume more operational time. After that, Mappers/Reducers just need the keys to en/decrypt the transmitted data. Instead, in Kerberos, each Mapper and Reducer must apply and secure the tickets, and AS needs to calculate the common secret key for TGS and Mappers/Reducers. Therefore, the more participators join the communications, the more operational time needs. Additionally, the secure data transmissions are additional loads. As shown in Fig. 13, the simulation results indicate that our proposed scheme outperforms Kerberos on secure data transmissions.

From the above simulation results, even if the Map/Reduce operation with threshold and group signature schemes, our proposed mechanism still performs well. Additionally, this study increases the number of tasks, Mappers and Reducers. The executive time of a Map/Reduce operation still raises

reasonably, and keeps a fine service level [2,15,26]. Therefore, our proposed mechanism is well suited for virtual cloud computing environments.

6 Conclusions and future work

Nowadays, cloud computing turns not into a tendency, also advancing at an unprecedented rate. Internet service provider (ISP) takes advantage of cloud computing to provide users internet services [18,19]. However, users still concern about the security issues. Because the computers of users and enterprises are located in cloud, and arbitrary clients can randomly log on and take private data away. As a result, the users are unaware of the functions of Map and Reduce likely disclose the valuable personal information. Consequently, the users and enterprise consider the insecure reasons, which influence the willing of users implementing applications in cloud environments. Hence, how to provide efficient and secure map/reduce functions is extremely essential.

In this study, we propose a secure Map/Reduce mechanism with fault tolerance ability. The proposed mechanism can rapidly locate vicious assaults, validate the delivered data integrity, and keep from impersonation attacks. Moreover, we adopt a digital signature mechanism to validate the identification of the Mapped/ Reducer and PKI VM. Also, the threshold crypto sharing scheme is exploited to defend and validate the correctness of delivered messages [20,21]. Finally, this study adopts virtual machines to perform and simulate the cloud computing environment. Simulation results demonstrate that our secured map/reduce mechanism only consumes a few computing power to execute secure functions. Therefore, the proposed mechanism is very appropriate for cloud environments. As future work, we will apply the proposed scheme to deal with attacks in cloud environments or various networks [27,28] to strengthen the platform security.

Acknowledgments This work was partly supported by National Science Council (NSC), Taiwan, under research Grants number NSC102-2221-E-126-003, NSC100-2221-E-126-006, and NTHU-Delta collaborative research projects. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Council, NTHU or Delta Electronics, Inc.

References

1. Creese, S., Hopkins, P., Pearson, S., & Shen, Y. (2009). Data protection-aware design for cloud services. *Lecture Notes in Computer Science*, vol. 5931, pp. 119–130. Berlin: Springer.
2. Iqbal, W., Dailey, M., & Carrera, D. (2010) SLA-driven automatic bottleneck detection and resolution for read intensive multitier applications hosted on a cloud. In: *Proceedings of the International Conference on Grid and Pervasive Computing*, pp. 37–46.
3. Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified data processing on large clusters. In: *Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation*.
4. Devanbu, P., Gertz, M., Stuart, C. M., & Stubblebine, G. (August 2000). Authentic third-party data publication. In: *Proceedings of the 14th IFIP 11.3 Working Conference in Database Security* .
5. Mackey, G., Sehrish, S., Lopez, J., Bent, J., Habib, S., & Wang, J. (2008). Introducing map reduce to high end computing. In: *Proceedings of the Petascale Data Storage Workshop Held in Conjunction with SC08*.
6. Ekanayake, J., Pallickara, S., & Fox, G. (2008). Mapreduce for data intensive scientific analysis. In: *Proceedings of the IEEE Fourth International Conference on eScience'08*, pp. 277–284. IEEE
7. Itani, W., Kayssi, A., & Chehab, A. (2009). Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures. In: *the Proceeding of the 8th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp. 711–716. IEEE
8. Begnum, K. (2010). Simplified cloud-oriented virtual machine management with MLN. *The Journal of Supercomputing*, *61*(2), 251-266. doi:10.1007/s11227-010-0424-0.
9. Yan, L., Rong, C., & Zhao, G. (2010). Strengthen Cloud Computing Security with Federal Identity Management Using Hierarchical Identity-Based Cryptography. *Lecture Notes in Computer Science*, vol. 5931. Berlin: Springer
10. Sun, H., & Aida, K. (2010). A hybrid and secure mechanism to execute parameter survey applications on local and public cloud resources. In: *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 118–126. IEEE
11. Zhou, D., Zhong, L., Wo, T., & Kang, J. (2010) CloudView: Describe and maintain resource view in cloud. In: *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 151–158. IEEE
12. Miyamoto, T., Hayashi, M., & Nishimura, K. (2010). Sustainable network resource management system for virtual private clouds. In: *Proceedings of the IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 512–520. IEEE
13. Desmedt, Y., & Frankel, Y. (August 1990). Threshold Crypto-Systems. In *Advances in Cryptology (Crypto'89)*, pp. 307–315. New York: Springer.
14. Stinson, D. R., & Wei, R. (August 1999). Unconditionally secure proactive secret sharing scheme with combinatorial. In: *Proceeding of the 6th Annual International Workshop Selected Areas in Cryptography*.
15. Wang, G. L. (2003). Security Analysis of Several Group Signature Schemes. *Lecture Notes in Computer Science*, vol. 2904, pp. 252–265. Berlin: Springer.
16. Fouquet, M., Hoene, C., Schläger, M., & Carle, G. (2011). Data collection in future mobile networks. *Telecommunication Systems Journal*, *48*(3–4), 289–300.
17. Altman, E., Ayesta, U., & Prabhu, B. J. (2011). Load balancing in processor sharing systems. *Telecommunication Systems Journal*, *47*(1–2), 35–48.
18. Ibrahiem, M. M., Emary, E., & Hassanien, A. E. (2011). Intelligent agent in telecommunication systems. *Telecommunication Systems Journal*, *46*(3), 191–193.
19. Hu, J., Deng, J., & Wu, J. (2013) A green private cloud architecture with global collaboration. *Telecommunication Systems Journal*, *52*(2), 1269–1279 doi:10.1007/s11235-011-9639-5.
20. Pearson, S., Shen, Y., & Mowbray, M. (2009). *A Privacy Manager for Cloud Computing* (pp. 90–106). Lecture Notes in Computer Science Berlin: Springer.
21. Elnikety, E., Elsayed, T., & Ramadan, H .E. (2011). iHadoop: Asynchronous iterations for MapReduce. In: *Proceedings of the 3rd*

IEE International Conference on Cloud Computing Technology and Science. IEEE
22. Chen, R., Gui, Y., & Gao, J. (2004). *Modification on Kerberos Authentication Protocol in Grid Computing Environment* (pp. 1079–1082). Lecture Notes in Computer Science Berlin: Springer.
23. Juan, W., Heng, C. M., & Kang, F. Y. (2011). An improved kerberos intra-domain authentication protocol based-on certificateless public-key cryptography. *Advances in Intelligent and Soft Computing*, *129*, 489–496.
24. Lin, H.-Y., Yang, C.-Y., & Hsieh, M.-Y. (2012). Secure map reduce data transmission mechanism in cloud computing using threshold secret sharing Schemes. In *Advances in Intelligent and Soft Computing*. Berlin: Springer
25. Hsieh, M.-Y., Lin, H.-Y., Lai, C.-F., & Li, K.-C. (2011). Secure protocols for data propagation and group communication in vehicular networks. *EURASIP Journal on Wireless Communications and Networking*, *2011*(1), 1–16.
26. Lin, H.-Y., & Chiang, T.-C. (2011). Efficient key agreements in dynamic multicast height balanced tree for secure multicast communications in Ad Hoc networks. *EURASIP Journal on Wireless Communications and Networking*, *2011*(1), 382701.
27. Hsieh, M.-Y. (2011). Data aggregation model using energy-efficient delay scheduling in multi-hop hierarchical wireless sensor networks. *IET Communications*, *5*(18), 2703–2711.
28. Yeh, C.-H., Hsieh, M.-Y., & Li, K.-C. (2014). *An Anonymous Communication Scheme with Non-reputation for Vehicular Ad Hoc Networks* (pp. 563–568). Lecture Notes in Electrical Engineering. Berlin: Springer.

Hua Yi Lin received the Ph.D. degree in Engineering Science from the National Cheng Kung University, Taiwan, in 2006. He is currently an assistant professor in Dept. of Management Information System at the China University of Technology, Taiwan, and a professional member of ACM and a member of IEEE. Most of his research areas are around ad-hoc networks (mobile routing protocol, secure routing protocol, QoS), sensor networks (secure data transmission, key management), network security, network planning, and multimedia MPEG. Dr. Lin served on program committees and editorial boards for several international conferences and journals.

Meng-Yen Hsieh received his MS degree in Computer Science & Information Engineering from National Central University, Taiwan, R.O.C. in 2001, and Ph.D. degree in Engineering Science from National Cheng Kung University, Taiwan, in 2007. He is currently an assistant professor of the Department of Computer Science & Information Engineering, Providence University, Taiwan. His research interests include mobile ad hoc networks, wireless sensor networks, wireless security, and software engineering. Dr. Hsieh served on symposium chairs and technical program committees for several international conferences.

Kuan-Ching Li is currently a Professor in the Department of Computer Science and Information Engineering at the Providence University, Taiwan. Dr. Li has published more than 120 papers in journals, books and conference proceedings. He has served in a number of journal editorial boards and guest editorship, as also served many international conference chairmanship positions as steering committee, advisory committee, general and program committee chairs and member of program committees. His research interests include networked computing (Clusters, Grids, P2Ps, Clouds), parallel software design, and performance evaluation and benchmarking. He is a senior member of the IEEE and a Fellow of the IET.