Flexible Access Control over Verifiable Cloud Computing Services with Provable Security

Ya-Fen CHANG

Department of Computer Science and Information Engineering
National Taichung University of Science and Technology, Taichung, Taiwan
e-mail: cyf@cs.ccu.edu.tw

Received: July 2013; accepted: March 2014

Abstract. This paper proposes an access control mechanism of verifiable cloud computing services using chameleon hashing and Diffie–Hellman key exchange protocol. By this mechanism, an entity can apply for cloud computing services and he can authorize other users to access granted data or services. When an authorized user or entity wants to access cloud computing services, he can authenticate the cloud computing service provider. Moreover, no entity secret will be revealed by data kept by cloud servers such that security and cost saving can be both ensured. Security proof under the simulation paradigm is also given.

Key words: cloud computing service, cloud computing, access control, chameleon hashing, Diffie–Hellman key exchange.

1. Introduction

Before the concept of cloud computing is introduced, similar concepts or computing models have been proposed such as distributed computing, grid computing, and utility computing. No matter which kind of computing model it is, the main concept is utilizing resources centrally. Cloud computing is composed of a front end, cloud computing services, and a back end, cloud computing technologies. Several institutes and companies, such as NIST, Gartner, Forrester, Google, Microsoft, IBM, and Wikipedia, define cloud computing with different definitions. But, could computing must provide the following four properties: (1) Services are provided via the Internet. (2) Resources can be managed and arranged dynamically. (3) A distributed virtual architecture exists. (4) Possible fees may be charged according to services requested or demands.

Cloud computing services can be divided into three categories: (1) Software as a Service (SaaS), (2) Platform as a Service (PaaS), and (3) Infrastructure as a Service (IaaS). Via SaaS, software is kept and maintained by SaaS providers, and users directly use it via the Internet. Via PaaS, customers can release their designed applications on the platform, such as Google and AppEngine. Via IaaS, customers can control operation systems, storages, networks, and applications without managing the infrastructure. According to deployment mechanisms, cloud computing services can be divided into three categories: (1) public cloud, (2) private cloud, and (3) hybrid cloud. With public cloud, services,
which users apply for, are provided by independent cloud computing service providers. Meanwhile, cloud computing service providers also provide services for other users. That is, resources of these cloud computing service providers are shared by users. With private cloud, the cloud computing infrastructure and environment are built and used by one specific company or institute. Users of private cloud must be internal ones such that external users cannot use cloud computing services of private cloud. With hybrid cloud, a company or institute has possessed a well-constructed system. The company only needs to classify data by security requirements and spends a little money to upgrade the existing system to utilize cloud computing services of flexibility, scalability, and high efficiency. No matter which kind of cloud computing service model is applied, it always can be built by the above three deployment mechanisms. Moreover, a user can freely choose a cloud computing service model with one specific deployment mechanism according to his security requirements.

Wireless technologies and systems, such as WLAN, WMAN, GPRS, and 3.5 G, provide ubiquitous network services. Users can use various kinds of devices to access cloud computing services via the Internet. Meanwhile, enterprises can save lots of costs to construct or maintain information infrastructure while utilizing cloud computing services provided by the cloud computing service provider. General users, small and medium-sized enterprises and venture companies tend to choose cloud computing services of public cloud such as Gmail and Google docs. On the other hand, financial institutions, government institutions and large-sized enterprises tend to choose cloud computing services of private cloud or hybrid cloud. However, whether the provided cloud computing services are under proper protection is still doubted. Thus, no matter which kind of deployment models the chosen cloud computing services belong to, it is always a tradeoff between security and cost saving.

In general, resources such as data and programs are kept by distributed systems to make them accessed easily. There usually exist many entities in distributed systems, where an entity may be a user or a group. Entities possess different rights on resources, and one entity only can access authorized resources according to the corresponding access right via access control (Denning et al., 1986; Denning, 1984; Davida et al., 1981). According to relationships between different entities, access control can be divided into two categories: (1) hierarchical access control and (2) arbitrary complicated access control. Because many organizations or companies are organized hierarchically, hierarchical access control is the most common (Chien and Jan, 2003; Chang et al., 1992; Hwang, 2000; Chung et al., 2008; Lin et al., 2003; Yeh et al., 1998; Chang and Chang, 2007). Arbitrary complicated access control provides flexible organization, but it is hard to be designed and managed. Thus arbitrary complicated access control is not common. Hierarchical access control can be further divided into two types: (1) conventional tree hierarchy access control (Chien and Jan, 2003; Chang et al., 1992; Hwang, 2000; Chung et al., 2008) and (2) complicated access control in a hierarchy (Lin et al., 2003; Yeh et al., 1998; Chang and Chang, 2007).

In general, access control on cloud computing services uses password and identity to authenticate users – Dropbox, Gmail, Amazon EC2, Apple iCould, and Amazon Cloud Storage for example. However, this approach makes cloud servers need to store a user’s
identity, password, and the corresponding encryption/decryption related information. If a customer wants to apply specific security services such as hierarchical access control to cloud computing services, common access control mechanisms cannot comply with this special requirement. The only way to solve this problem is customization or mechanism integration. Nevertheless, this approach tends to be time-consuming and expensive.

In this paper, the concepts of chameleon hashing (Krawczyk and Rabin, 2000) and Diffie–Hellman key exchange protocol are employed to propose an access control mechanism for verifiable cloud computing services. In the proposed access control mechanism, an entity can apply for cloud computing services via the trustworthy registration center’s help. The trustworthy registration center acts as a cloud computing service broker, and all users and cloud computing service providers trust the registration center (Chonka et al., 2011). The entity can authorize other users to access granted data or services by issuing them corresponding authorization items. Thereupon, a user can use the issued authorization items to access cloud computing services. Meanwhile, cloud computing service providers can be authenticated, and cloud computing service providers cannot obtain any secret information of entities or users. Moreover, the security is proved under the simulation paradigm.

The rest of this paper is organized as follows. Section 2 introduces chameleon hashing. The proposed access control mechanism for verifiable cloud computing services is shown in Section 3 followed by analyses in Section 4. At last, some conclusions are made in Section 5.

2. Chameleon Hashing

In this section, the concept of chameleon hashing is introduced. Krawczyk and Rabin first proposed the concept of chameleon hashing (2000) and proposed chameleon signature by chameleon hashing. Chameleon hashing is different from other collision-resistant hashing such as SHA-1. Chameleon hashing possesses four properties. Before demonstrating these four properties, suppose there exist a key pair \((pk_{hash}, sk_{FindingCollisionTrapdoor})\) and a chameleon hash function \(H(.)\), where \(pk_{hash}\) is public and \(sk_{FindingCollisionTrapdoor}\) is private.

**Computation efficiency:** If \(pk_{hash}\) is known, it is efficient to compute \(H_{pk_{hash}}(m, r)\) with given \(m\) and \(r\).

**Collision resistance:** When \(sk_{FindingCollisionTrapdoor}\) is unknown, it is hard to find \((m_1, r_1)\) and \((m_2, r_2)\) such that \(m_1 \neq m_2\) and \(H_{pk_{hash}}(m_1, r_1) = H_{pk_{hash}}(m_2, r_2)\).

**Trapdoor collision:** When \(sk_{FindingCollisionTrapdoor}\) is known, there exists an efficient algorithm to get \((m_2, r_2)\) with given \((m_1, r_1)\) and \(pk_{hash}\) such that \(H_{pk_{hash}}(m_1, r_1) = H_{pk_{hash}}(m_2, r_2)\).

**Uniformity:** With arbitrary \(m\), \(H_{pk_{hash}}(m, r)\) reveals no information related to it because \(r\) is chosen uniformly.

Krawczyk and Rabin (2000) proposed chameleon hashing based on the difficulty of solving discrete logarithm problems. The proposed chameleon hash is \(H(m, r) = g^m y^r \mod p\), where \(p\) is a large prime, \(g\) is a generator in \(GF(p)\), \(x\) is the private key, and \(y = g^x \mod p\) is the public key.
3. The Proposed Access Control Mechanism for Verifiable Cloud Computing Services

In this section, the proposed access control mechanism for verifiable cloud computing services is demonstrated. There exists one trustworthy registration center $R$ in the access control mechanism. The trustworthy registration center $R$ is responsible for transmitting essential data to a cloud computing service provider $S_j$ when an entity $G_i$ applies for $S_j$’s service for the first time. Both $S_j$ and $G_i$ trust $R$, and an entity $G_i$ can be a group, a company, an institution, or a user. The proposed access control mechanism is composed of four phases: cloud computing service applying phase, authorization verification item adding and updating phase, authorization item obtaining phase, and cloud computing service accessing phase. The details are as follows.

### 3.1. Cloud Computing Service Applying Phase

In this phase, an entity $G_i$, who wants to apply for a cloud computing service provider $S_j$’s cloud computing service, needs to register at $S_j$ with the trustworthy registration center $R$’s assistance. In this phase, data must be delivered via secure channels. This phase is depicted in Fig. 1, and the details are as follows.

**Step 1:** $G_i$ chooses his identity $GID_i$ and sends a registration request including $GID_i$ to $R$.

**Step 2:** After getting $G_i$’s registration request, $R$ checks if the received $GID_i$ is duplicated. If it is fresh, $R$ chooses a large prime $p$, a primitive element $g$ in $GF(p)$ and a collision-resistant hash function $h(.)$ and sends $\{g, p, h(.)\}$ to $G_i$, where $p = 2q + 1$ and $q$ is a large prime.

**Step 3:** After receiving $\{g, p, h(.)\}$, $G_i$ randomly chooses $x_i \in Z_{p−1}$ and $\beta_i \in Z_{p−1}$ and computes $y_i = g^{xi} \mod p$, $A_i = g^{\beta_i} \mod p$, and $s_i = h(x_i || A_i || \beta_i)$, where $||$ denotes a concatenation operator. At last, $G_i$ stores $\{GID_i, x_i, \beta_i, g, p, h(.)\}$ and sends $\{GID_i, A_i, y_i, s_i\}$ to $R$.
Step 4: When receiving \([GID_i, A_i, y_i, s_i, g, p, h(.)]\), \(R\) sends \([GID_j, A_j, y_j, s_j, g, p, h(.)]\) to \(S_j\). At last, \(S_j\) stores an authentication entry \([GID_j, A_j, y_j, s_j, g, p, h(.)]\) to authenticate \(G_i\).

3.2. Authorization Verification Item Adding and Updating Phase

After applying for \(S_j\)'s service, \(G_i\) can add or update authorization verification items such that only users authorized by \(G_i\) can access the cloud computing service which \(S_j\) provides to \(G_i\). When \(G_i\) only wants to update authorization verification items, all \(G_i\) needs to do is explicitly indicating them. In the following, how \(G_i\) adds a new authorization verification item is shown, which is depicted in Fig. 2.

**Step 1:** When \(G_i\) wants to add a new authorization verification item, \(G_i\) sends an authorization verification item adding request with \(GID_i\) and \(A_i\) to \(S_j\).

**Step 2:** After getting the authorization verification item adding request, \(S_j\) checks if there exists a record with respect to \(GID_i\) and \(A_i\). If such an item exists, \(S_j\) chooses three random numbers \(\alpha \in Z_{p-1}^*, \omega \in Z_{p-1}, x_{ik} \in Z_{p-1}\) and computes \(m_1 = g^\alpha \mod p\), \(y_{ik} = g^{x_{ik}} \mod p\), \(m_2 = A_i^{x_{ik}} \mod p\), and \(m_3 = h(m_1 || \omega || s_i || y_i || m_2)\). Then, \(S_j\) sends \([m_1, m_3, y_{ik}, \omega]\) to \(G_i\).
Step 3: After getting $S_j$’s response $\{m_1, m_3, y_{ik}, \omega\}$, $G_i$ computes $m_4 = h(x_i || A_i || \beta_{ik})$ and $m_5 = y_{ik}^{\beta_{ik}} \mod p$ and checks if $m_3 = h(m_1 || \omega || m_4 || g^\gamma) \mod p || m_5$ holds or not. If it does not hold, $G_i$ regards $S_j$ as an illegal cloud computing service provider and terminates the phase immediately. Otherwise, $G_i$ computes $r = \beta_i - x_i \omega \mod (p - 1)$, $m_6 = m'_i \mod p$, and $m_7 = g^{r \times m_4} \mod p$. $G_i$ randomly chooses $\beta_{ik} \in Z_{p-1}$ and computes $A_{ik} = g^{\beta_{ik}} \mod p$, $s_{ik} = h(x_i || A_{ik} || \beta_{ik})$, $m_8 = A_{ik} \oplus h(m_7)$, $m_9 = s_{ik} \oplus m_7$, and $m_{10} = h(A_{ik} || s_{ik} || m_4)$. At last, $G_i$ stores $\{A_{ik}, \beta_{ik}, y_{ik}\}$ and sends $\{GID_i, A_i, m_6, m_8, m_9, m_{10}\}$ to $S_j$.

Step 4: After getting $\{GID_i, A_i, m_6, m_8, m_9, m_{10}\}$, $S_j$ computes $m_{11} = m_6^{\omega} \mod p$ and checks if $A_i = m_{11} \times y_{ik}^{\omega} \mod p$ holds or not. If it does not hold, $S_j$ regards $G_i$ as an illegal entity and terminates this phase immediately. Otherwise, $S_j$ computes $m_{12} = m_1 \mod p$, $A'_i = m_8 \oplus h(m_{12})$, and $s_{ik}' = m_9 \oplus m_{12}$ and checks if $m_{10} = h(A'_i || s_{ik}' || s_i)$ hold or not. If they are not equal, $S_j$ regards $G_i$’s authentication data is incorrect and denies $G_i$’s authorization verification item adding request; otherwise, $S_j$ stores the new-added authorization verification item $\{A'_{ik}, s_{ik}', y_{ik}\}$. Note that $S_j$ may store multiple authorization verification items for one entity $G_i$.

3.3. Authorization Item Obtaining Phase

In this phase, a user $U$ can send a request to $G_i$ to get authorization items when $U$ wants to get $G_i$’s data or share $G_i$’s resource. $G_i$ can determine which data or resource can be accessed by $U$. In this phase, data must be delivered via secure channels. This phase is depicted in Fig. 3, and the details are as follows.

Step 1: $U$ sends an authorization item obtaining request to $G_i$.

Step 2: $G_i$ determines which data or resource can be accessed by $U$ and gets appropriate $\{A_{ik}, \beta_{ik}, y_{ik}\}$. At last, $G_i$ sends $\{GID_i, x_i, A_{ik}, \beta_{ik}, y_{ik}, g, p, h(.)\}$ to $U$. Note that $U$ may be issued multiple $\{A_{ik}, \beta_{ik}, y_{ik}\}$’s.

3.4. Cloud Computing Service Accessing Phase

After getting $G_i$’s authorization items, $U$ can directly access $G_i$’s corresponding data or service provided by $S_j$. When $U$ wants to access $S_j$’s service, this phase is depicted in
Fig. 4 and the details are as follows. If \( G_i \) wants to access \( S_j \)'s service, he acts as \( U \) in this phase.

**Step 1:** \( U \) chooses a random number \( \sigma \in \mathbb{Z}_{p-1} \), computes \( m_1 = g^\sigma \mod p \), and sends \( GID_i, A_{ik} \) and \( m_1 \) to \( S_j \) as a cloud computing service access request.

**Step 2:** After getting the request, \( S_j \) first checks if there exists an entry with respect to \( GID_i \) and \( A_{ik} \). If no such entry exists, \( S_j \) terminates this phase directly. Otherwise, \( S_j \) chooses two random numbers \( \alpha \in \mathbb{Z}_{p-1} \) and \( \omega \in \mathbb{Z}_{p-1} \), computes \( m_2 = g^\alpha \mod p \), \( m_3 = m_1^\alpha \mod p \), \( m_4 = h(x_i) || A_{ik} || \beta_{ik} \mod p \), \( m_5 = m_3^\omega \mod p \), \( m_6 = y_{ik}^\alpha \mod p \), \( m_7 = m_2^\omega \mod (p-1) \), \( m_8 = m_2^\alpha \mod p \), \( m_9 = m_7^\alpha \mod (p-1) \), \( m_{10} = m_1^\alpha \mod p \), \( m_6 = y_{ik}^\alpha \mod p \), \( m_7 = m_2^\omega \mod (p-1) \), \( m_8 = m_2^\alpha \mod p \), \( m_9 = m_7^\alpha \mod (p-1) \), \( m_{10} = m_1^\alpha \mod p \), \( A_{ik} \in \mathbb{Z}_{p-1} \), and sends \( GID_i, A_{ik}, m_7 \) to \( U \).

**Step 3:** After receiving \( \{m_2, m_4, \omega\} \), \( U \) computes \( m_5 = h(x_i) || A_{ik} || \beta_{ik} \mod p \) and \( m_6 = y_{ik}^\alpha \mod p \) and checks if \( m_4 = h(m_2 || \omega || m_5 \mod p) || m_6 \). If it does not hold, \( U \) regards \( S_j \) as an illegal cloud computing service provider and terminates this phase immediately; otherwise, \( U \) computes \( r = \beta_{ik} - x_i \omega \mod (p-1) \), \( m_7 = m_5^\alpha \mod p \), \( m_8 = m_3^\omega \mod p \), and \( m_9 = m_2^\alpha \mod p \). \( U \) sends \( GID_i, A_{ik}, m_7 \) to \( S_j \).

**Step 4:** After getting the reply \( \{GID_i, A_{ik}, m_7\} \), \( S_j \) computes \( m_9 = m_7^\alpha \mod p \) and checks if \( A_{ik} = m_9 \times y_{ik}^\omega \mod p \). If it does not hold, \( S_j \) regards \( U \) as an illegal user and terminates this phase immediately; otherwise, \( S_j \) computes \( m_{10} = m_1^\alpha \times s_{ik} \mod p \).

After the above steps, \( S_j \) and \( U \) authenticate each other and \( m_8 = m_{10} = g^{\alpha \times \sigma \times s_{ik}} \mod p \). The session key, \( m_8/m_{10} \), can be used to protect the following communication content.

In this section, security analyses are first given to show that no one can retrieve unauthorized secret and impersonate one party to cheat another legitimacy party. Then further discussions are made to show properties of the proposed mechanism.

4.1. Security Analyses

The security bases of the proposed mechanism are secure one-way hash function and discrete logarithm problem. The corresponding definitions are given as follows.

**Definition 1.** A one-way hash function, \( h(.) : x \rightarrow y \), is secure if it is easy to compute \( h(x) = y \) of fixed length for any given variable \( x \) while it is computationally infeasible to derive \( x \) from any given \( y \).

**Definition 2.** Discrete logarithm problem (DLP): Let \( G \) be an abelian group, \( g \in G \) of order \( n \). Given \( y \in G \), find \( x \) such that \( y = g^x \).

**Theorem 1 (Euler’s Theorem).** If \( p \) and \( \alpha \) are coprime positive integers, then \( \alpha^{\phi(p)} \equiv 1 \pmod{p} \), where \( \phi(p) \) is Euler’s totient function.

**Lemma 1.** Let \( h(.) \) be a secure one-way hash function with outputs of fixed length \( \text{Len}_h \). There exists a negligible probability \( \frac{1}{2^{\text{Len}_h}} \) to find \( x \) from given \( y \) such that \( h(x) = y \).

**Proof.** Outputs of \( h(.) \) are of fixed length \( \text{Len}_h \) so there are \( \frac{1}{2^{\text{Len}_h}} \) patterns for \( h(.) \) outputs. For a variable \( x \) and given \( y \), the probability of \( h(x) = y \) is \( \frac{1}{2^{\text{Len}_h}} \). If \( h(.) \) is a secure one-way hash function, it is computationally infeasible to derive \( x \) from any given \( y \) such that \( \frac{1}{2^{\text{Len}_h}} \) is negligible. \( \square \)

**Lemma 2.** If DLP is hard, there exists a negligible probability \( \frac{1}{p-1} \) to find \( x \) from given \( y \) such that \( y = g^x \pmod{p} \), where \( p \) is a prime and \( g \) is a primitive element in \( GF(p) \).

**Proof.** Because \( p \) is a prime, \( p \) and \( g \) are coprime. According to Theorem 1, \( g^{\phi(p)} \equiv g^{(p-1)} \equiv 1 \pmod{p} \). Since \( g \) is a primitive element in \( GF(p) \), the order for \( g \) is \( (p-1) \) such that \( x \) is in \([1, p-1]\). Given \( p, g, \) and \( y \), the probability of \( y = g^x \pmod{p} \) is \( \frac{1}{p-1} \). If DLP is hard, it is intractable to find \( x \) such that \( y = g^x \pmod{p} \) such that \( \frac{1}{p-1} \) is negligible. \( \square \)

In order to prove the security of the proposed mechanism, the concept from the simulation paradigm is taken. For any adversary \( A \), given \( g, p, \) and \( h(.) \), \( A \) is capable of intercepting transmitted data, randomly choosing \( \delta \), and computing \( g^\delta \pmod{p} \) and hash function operations. Within a time bound \( t \), \( A \) can determine whether the retrieved secret
Lemma 3. Let A be neither a user of G_i nor S_j. Given g, p, h(.) and the intercepted messages in authorization verification item adding and updating phase and cloud computing service accessing phase, A can retrieve G_i’s secret β_i, G_i’s authorization item secret β_{ik}, or S_j’s secret x_{ik} within a time bound t by making at most q_{passive} passive checking iterations of probability at most $1/p-1 + (1 - 1/p - 1)^{q_{passive}}$.

Proof. Let S_1 denote the event that A can retrieve G_i’s secret β_i, G_i’s authorization item secret β_{ik}, or S_j’s secret x_{ik} within a time bound t by making at most q_{passive} passive checking iterations. $Pr[S_1]$ denotes the successful probability of event S_1. A has g, p, h(.) and the intercepted messages transmitted via insecure channels. A is capable of randomly choosing $δ ∈ Z_{p-1}$ and computing $R = g^δ \mod p$. For each checking iteration, A randomly chooses $δ ∈ Z_{p-1}$ and computes $R = g^δ \mod p$. If R equals $A_i$, $β_i = δ$. If R equals $A_{ik}$, $β_{ik} = δ$. If R equals $S_j$, $G_i = δ$. From above, we have

$$Pr[S_1] = \frac{1}{p-1} + \sum_{i=2}^{q_{passive}} \left( \frac{1}{p-1} \prod_{j=2}^{i} \left( 1 - \frac{1}{p-j+1} \right) \right).$$

We can get the lower and upper bounds of $Pr[S_1]$ as follows:

$$\frac{1}{p-1} + \frac{(p - q_{passive} + 1)(1 - \left( \frac{1}{p - q_{passive} + 1} \right)^{q_{passive}})}{p - 2} \leq Pr[S_1], \quad (1)$$

$$Pr[S_1] \leq \frac{1}{p-1} + \frac{(p-1)(1 - \left( 1 - \frac{1}{p} \right)^{q_{passive}})}{p - q_{passive}}. \quad (2)$$

Because $p$ is a large prime, Eqs. (1) and (2) are rewritten as follows.

$$\frac{1}{p-1} + \left( 1 - \left( 1 - \frac{1}{p - q_{passive} + 1} \right)^{q_{passive}} \right) \leq Pr[S_1]$$

$$\leq \frac{1}{p-1} + \left( 1 - \left( 1 - \frac{1}{p-1} \right)^{q_{passive}} \right). \quad (3)$$

Lemma 4. Let A be a user of G_i. Given an authorization item $\{A_{ik}, β_{ik}, y_{ik}\}$, GID_t, x_i, g, p, h(.), and intercepted messages in authorization verification item adding and updating
phase and cloud computing service accessing phase. A can retrieve $G_i$'s secret $\beta_i$ or $S_j$'s secret $x_{ik}$ within a time bound $t$ by making at most $q_{\text{passive}}$ passive checking iterations of probability at most $\frac{1}{p-1} + (1 - \frac{1}{p-1})^{q_{\text{passive}}}$.

Proof. Let $S_2$ denote the event that $A$ can retrieve $G_i$'s secret $\beta_i$ or $S_j$'s secret $x_{ik}$ within a time bound $t$ by making at most $q_{\text{passive}}$ passive checking iterations. $Pr[S_2]$ denotes the successful probability of event $S_2$. $A$ has an authorization item $\{A_{ik}, \beta_{ik}, y_{ik}\}$, GID$_i$, $x_i$, $g$, $p$, $h(.)$ and the intercepted messages transmitted via insecure channels. $A$ is capable of randomly choosing $\delta \in \mathbb{Z}_{p-1}$ and computing $R = m^q \mod p$, where $m_1 = g^q \mod p$ intercepted in authorization verification item adding and updating phase. If $R$ and $m_6$ are equal, $r = \delta$ and $\beta_i = \delta + x_i \omega \mod (p - 1)$. From above, the probability to retrieve $G_i$'s secret $\beta_i$ by making at most $q_{\text{passive}}$ passive checking iterations is $\frac{1}{p-1} + \sum_{i=2}^{q_{\text{passive}}} \left(\frac{1}{p-1} \prod_{j=2}^{i} (1 - \frac{1}{p-j+1})\right)$. When $A$ randomly chooses $\delta \in \mathbb{Z}_{p-1}$ and computes $R = g^q \mod p$, $A$ checks if $R$ equals $y_{ik}$. If it holds, $x_{ik} = \delta$. Thus, the probability to retrieve $S_j$'s secret $x_{ik}$ by making at most $q_{\text{passive}}$ passive checking iterations is $\frac{1}{p-1} + \sum_{i=2}^{q_{\text{passive}}} \left(\frac{1}{p-1} \prod_{j=2}^{i} (1 - \frac{1}{p-j+1})\right)$. From above, we have

$$Pr[S_2] = \frac{1}{p-1} + \sum_{i=2}^{q_{\text{passive}}} \left(\frac{1}{p-1} \prod_{j=2}^{i} (1 - \frac{1}{p-j+1})\right).$$

The lower and upper bounds of $Pr[S_2]$ are as follows:

$$\frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p-q_{\text{passive}}+1}\right)^{q_{\text{passive}}^{-1}}\right) \leq Pr[S_2] \leq \frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p-1}\right)^{q_{\text{passive}}^{-1}}\right). \quad (4)$$

Lemma 5. Let $A$ be a user of $G_i$. Given an authorization item $\{A'_{ik}, \beta'_{ik}, y'_{ik}\}$, GID$_i$, $x_i$, $g$, $p$, $h(.)$, and intercepted messages in authorization verification item adding and updating phase and cloud computing service accessing phase, $A$ can retrieve another authorization item $\{A_{ik}, \beta_{ik}, y_{ik}\}$ of $G_i$ by making at most $q_{\text{passive}}$ passive checking iterations of probability at most $\frac{1}{p-1} + (1 - \frac{1}{p-1})^{q_{\text{passive}}^{-1}}$.

Proof. Let $S_3$ denote the event that $A$ can retrieve another authorization item $\{A_{ik}, \beta_{ik}, y_{ik}\}$ of $G_i$ within a time bound $t$ by making at most $q_{\text{passive}}$ passive checking iterations. $Pr[S_3]$ denotes the successful probability of event $S_3$. $A$ has an authorization item $\{A'_{ik}, \beta'_{ik}, y'_{ik}\}$, GID$_i$, $x_i$, $g$, $p$, $h(.)$ and the intercepted messages transmitted via insecure channels. $A$ is capable of randomly choosing $\delta \in \mathbb{Z}_{p-1}$ and computing $R = m^q \mod p$, where $m_2 = g^q \mod p$ intercepted in cloud computing service accessing phase. If $R$ and $m_7$ are equal, $r = \delta$ and $\beta_{ik} = \delta + x_i \omega \mod (p - 1)$. From above, the probability to retrieve
another authorization item \( \{A_{ik}, \beta_{ik}, y_{ik}\} \) by making at most \( q_{\text{passive}} \) passive checking iterations is \( \frac{1}{p^{-1}} + \sum_{i=2}^{q_{\text{passive}}} \left( \frac{1}{p^{-1}} \prod_{j=2}^{i} \left( 1 - \frac{1}{p^{-j+1}} \right) \right) \). From above, we have

\[
\Pr[S3] = \frac{1}{p^{-1}} + \sum_{i=2}^{q_{\text{passive}}} \left( \frac{1}{p^{-i}} \prod_{j=2}^{i} \left( 1 - \frac{1}{p^{-j+1}} \right) \right).
\]

The lower and upper bounds of \( \Pr[S3] \) are as follows:

\[
\frac{1}{p^{-1}} + \left( 1 - \left( 1 - \frac{1}{p^{-q_{\text{passive}}+1}} \right) \right) \left( q_{\text{passive}}^{-1} \right) \\
\leq \Pr[S3] \leq \frac{1}{p^{-1}} + \left( 1 - \left( 1 - \frac{1}{p^{-1}} \right) \right) \left( q_{\text{passive}}^{-1} \right). \quad (5)
\]

**Lemma 6.** Let \( A \) be \( S_j \). Given GID, \( A_i, y_i, s_i, g, p, h(.) \), \( \{A_{ik}, s_{ik}, x_{ik}\} \), and intercepted messages in authorization verification item adding and updating phase and cloud computing service accessing phase, \( A \) can retrieve \( G_i \)’s secret \( \beta_i \) or \( G_i \)’s authorization item secret \( \beta_{ik} \) within a time bound \( r \) by making at most \( q_{\text{passive}} \) passive checking iterations of probability at most \( \frac{1}{p^{-1}} + (1 - \left( 1 - \frac{1}{p^{-1}} \right)) q_{\text{passive}}^{-1} \).

**Proof.** Let \( S4 \) denote the event that \( A \) can retrieve \( G_i \)’s secret \( \beta_i \) or \( G_i \)’s authorization item secret \( \beta_{ik} \) within a time bound \( r \) by making at most \( q_{\text{passive}} \) passive checking iterations. \( \Pr[S4] \) denotes the successful probability of event \( S4 \). \( A \) has GID, \( A_i, y_i, s_i, g, p, h(.) \), \( \{A_{ik}, s_{ik}, x_{ik}\} \) and the intercepted messages transmitted via insecure channels. \( A \) is capable of randomly choosing \( \delta \in Z_{p^{-1}} \) and computing \( R = g^\delta \mod p \). For each checking iteration, \( A \) randomly chooses \( \delta \in Z_{p^{-1}} \) and computes \( R = g^\delta \mod p \). If \( R \) equals \( A_i, \beta_i = \delta \). If \( R \) equals \( A_{ik}, \beta_{ik} = \delta \). From above, we have

\[
\Pr[S4] = \frac{1}{p^{-1}} + \sum_{i=2}^{q_{\text{passive}}} \left( \prod_{j=2}^{i} \left( 1 - \frac{1}{p^{-j+1}} \right) \right)
\]

Because \( p \) is a large prime, the lower and upper bounds of \( \Pr[S4] \) are as follows:

\[
\frac{1}{p^{-1}} + \left( 1 - \left( 1 - \frac{1}{p^{-q_{\text{passive}}+1}} \right) \right) \left( q_{\text{passive}}^{-1} \right) \\
\leq \Pr[S4] \leq \frac{1}{p^{-1}} + \left( 1 - \left( 1 - \frac{1}{p^{-1}} \right) \right) \left( q_{\text{passive}}^{-1} \right). \quad (6)
\]

**Lemma 7.** Let \( A \) be \( G_i \). Given GID, \( x_i, A_i, \beta_i, g, p, h(.) \), \( \{A_{ik}, \beta_{ik}, y_{ik}\} \), and intercepted messages in authorization verification item adding and updating phase and cloud computing service accessing phase, \( A \) can retrieve \( S_j \)’s secret \( x_{ik} \) within a time bound \( r \).
bound $t$ by making at most $q_{\text{passive}}$ passive checking iterations of probability at most 
\[ \frac{1}{p-1} + \left(1 - \frac{1}{p} q_{\text{passive}}\right). \]

Proof. Let $S5$ denote the event that $A$ can retrieve $S_j$’s secret $x_{ik}$ within a time bound $t$ by making at most $q_{\text{passive}}$ passive checking iterations. $Pr[S5]$ denotes the successful probability of event $S5$. $A$ has $GID_i, x_i, A_i, \beta_i, g, p, h(\cdot), [A_{ik}, \beta_{ik}, y_{ik}]$, and the intercepted messages transmitted via insecure channels. $A$ is capable of randomly choosing $\delta \in \mathbb{Z}_{p-1}$ and computing $R = g^\delta \pmod{p}$. $A$ checks if $R$ equals $y_{ik}$. If it holds, $x_{ik} = \delta$. Thus, the probability to retrieve $S_j$’s secret $x_{ik}$ by making at most $q_{\text{passive}}$ passive checking iterations is 
\[ \frac{1}{p-1} + \sum_{i=2}^{q_{\text{passive}}} \left( \frac{1}{p-1} \prod_{j=2}^{i} (1 - \frac{1}{p-1}) \right). \]
From above, we have 
\[ Pr[S5] = \frac{1}{p-1} + \sum_{i=2}^{q_{\text{passive}}} \left( \frac{1}{p-1} \prod_{j=2}^{i} (1 - \frac{1}{p-1}) \right). \]
The lower and upper bounds of $Pr[S5]$ are as follows:
\[ \frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p} q_{\text{passive}} + 1\right)^{-1} \right) \]
\[ \leq Pr[S5] \leq \frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p-1}\right)^{q_{\text{passive}}} \right). \]

Theorem 2. Suppose solving DLP is hard, no adversary $A$ can retrieve any unauthorized secret with a non-negligible probability.

Proof. By Lemmas 3–7, upper bounds for probabilities of retrieving unauthorized secret are given. If DLP is hard, the probability $\frac{1}{p-1}$ is negligible such that probabilities of retrieving any secret approximate 0. For any adversary $A$, the probability to retrieve unauthorized secret is negligible.

Lemma 8. Let $A$ know none of $s_i, \beta_i$, and $y_i$. In authorization verification item adding and updating phase, $A$ can impersonate $S_j$ to cheat $G_i$ within a time bound $t$ by making at most $q_{\text{active}}$ active authentication sessions of probability at most 
\[ \frac{1}{\text{Len}_h} + \left(1 - \left(1 - \frac{1}{p} q_{\text{active}}\right)^{\text{Len}_h} \right). \]

Proof. Let $S6$ denote the event that $A$ impersonates $S_j$ to cheat $G_i$ within a time bound $t$ by making at most $q_{\text{active}}$ active authentication sessions. $Pr[S6]$ denotes the successful probability of event $S6$. Without knowing $s_i, \beta_i$, and $y_i$, $A$ is capable of randomly choosing $\delta$ of length $\text{Len}_h$. After getting the authorization verification item adding request, $A$ chooses three random numbers $\alpha \in \mathbb{Z}_{p-1}, \omega \in \mathbb{Z}_{p-1}$, and $x_{ik} \in \mathbb{Z}_{p-1}$ and computes 
\[ m_1 = g^\alpha \pmod{p}, y_{ik} = g^\omega \pmod{p}, \text{ and } m_2 = A_{ik}^\delta \pmod{p}. \]
Because $A$ does not know $s_i, \beta_i$, and $y_i$, $A$ sends $[m_1, m_3, y_{ik}, \omega]$ to $G_i$, where $m_3 = \delta$. After getting $[m_1, m_3, y_{ik}, \omega]$, $G_i$ authenticates $A$ as mentioned in authorization verification item adding and updating.
phase. A can be authenticated successfully if and only if 
\[ \delta = h(m_1 || \omega || s_i || y_i || m_2). \]
From above, we have
\[
Pr[S6] = \frac{1}{2\text{Len}_h} + \sum_{i=2}^{q_{\text{active}}} \left( \frac{1}{2\text{Len}_h - i + 1} \prod_{j=2}^{i} \left( 1 - \frac{1}{2\text{Len}_h - j + 2} \right) \right).
\]
And bounds for \( Pr[S6] \) are as follows:
\[
\frac{1}{2\text{Len}_h} + \left( 1 - \frac{1}{2\text{Len}_h - q_{\text{active}} + 2} \right)^{q_{\text{active}}}
\leq Pr[S6] \leq \frac{1}{2\text{Len}_h} + \left( 1 - \frac{1}{2\text{Len}_h} \right)^{q_{\text{active}}}. \tag{8}
\]

**Lemma 9.** Let A be unaware of \( \beta_i \). In authorization verification item adding and updating phase, A can impersonate \( G_i \) to cheat \( S_j \) within a time bound \( t \) by making at most \( q_{\text{active}} \) active authentication sessions of probability at most \( \frac{1}{p-1} + (1 - \frac{1}{p-1})^{q_{\text{active}}} \).

**Proof.** Let \( S7 \) denote the event that A impersonates \( G_i \) to cheat \( S_j \) within a time bound \( t \) by making at most \( q_{\text{active}} \) active authentication sessions. \( Pr[S7] \) denotes the successful probability of event \( S7 \). Without knowing \( \beta_i \), A is capable of randomly choosing \( \delta \in Z_{p-1} \) and computing \( R = m_2^\delta \mod p \). A can be authenticated if and only if \( \delta = \beta_i - x_i \omega \mod (p - 1) \). From above, we have
\[
Pr[S7] = \frac{1}{p-1} + \sum_{i=2}^{q_{\text{active}}} \left( \frac{1}{p-1} \prod_{j=2}^{i} \left( 1 - \frac{1}{p - j + 1} \right) \right).
\]
We can get the lower and upper bounds of \( Pr[S7] \) as follows:
\[
\frac{1}{p-1} + \frac{(p - q_{\text{active}} + 1)(1 - (1 - \frac{1}{p - q_{\text{active}} + 1})^{q_{\text{active}}})}{p - 2} \leq Pr[S7],
\]
\[
Pr[S7] \leq \frac{1}{p-1} + \frac{(p - 1)(1 - (1 - \frac{1}{p})^{q_{\text{active}}})}{p - q_{\text{active}}}. \tag{9}
\]
Because \( p \) is a large prime, the lower and upper bounds of \( Pr[S7] \) are as follows:
\[
\frac{1}{p-1} + \left( 1 - \frac{1}{p - q_{\text{active}} + 1} \right)^{q_{\text{active}}}
\leq Pr[S7] \leq \frac{1}{p-1} + \left( 1 - \frac{1}{p-1} \right)^{q_{\text{active}}}. \tag{9}
\]
Lemma 10. Let $A$ be unaware of $x_{ik}$. In cloud computing service accessing phase, $A$ can impersonate $S_j$ to cheat $U$ within a time bound $t$ by making at most $q_{active}$ active authentication sessions of probability at most $\frac{1}{2\text{Len}_b} + (1 - \frac{1}{2\text{Len}_b})^{q_{active} - 1}$.

Proof. Let $S8$ denote the event that $A$ impersonates $S_j$ to cheat $U$ within a time bound $t$ by making at most $q_{active}$ active authentication sessions. $Pr[S8]$ denotes the successful probability of event $S8$. Without knowing $x_{ik}$, $A$ is capable of randomly choosing $\delta$ of length $\text{Len}_b$. After getting the cloud computing service accessing request, $A$ chooses random numbers $\alpha \in \mathbb{Z}_{p-1}^*$ and $\omega \in \mathbb{Z}_{p-1}$ and computes $m_2 = g^\alpha \mod p$. Because $A$ does not know $x_{ik}$, $A$ sends $[m_2, m_4, \omega]$ to $U$, where $m_4 = \delta$. After getting $[m_2, m_4, \omega]$, $U$ authenticates $A$ as mentioned in cloud computing service accessing phase. $A$ can be authenticated successfully if and only if $\delta = h(m_2||\omega||x_{ik}||y_i||m_1)$. From above, we have

$$Pr[S8] = \frac{1}{2\text{Len}_b} + \sum_{i=2}^{q_{active}} \left(2\text{Len}_b - i + 1\right) \prod_{j=2}^{i} \left(1 - \frac{1}{2\text{Len}_b} - j + 2\right).$$

And bounds for $Pr[S8]$ are as follows:

$$\frac{1}{2\text{Len}_b} + \left(1 - \left(1 - \frac{1}{2\text{Len}_b} - q_{active} + 2\right)^{q_{active} - 1}\right) \leq Pr[S8] \leq \frac{1}{2\text{Len}_b} + \left(1 - \left(1 - \frac{1}{2\text{Len}_b}\right)^{q_{active} - 1}\right). \quad (10)$$

Lemma 11. Let $A$ be unaware of $\beta_{ik}$. In cloud computing service accessing phase, $A$ can impersonate $U$ to cheat $S_j$ within a time bound $t$ by making at most $q_{active}$ active authentication sessions of probability at most $\frac{1}{p-1} + (1 - \frac{1}{p-1})^{q_{active} - 1}$.

Proof. Let $S9$ denote the event that $A$ impersonates $U$ to cheat $S_j$ within a time bound $t$ by making at most $q_{active}$ active authentication sessions. $Pr[S9]$ denotes the successful probability of event $S9$. Without knowing $\beta_{ik}$, $A$ is capable of randomly choosing $\delta \in \mathbb{Z}_{p-1}$ and computing $R = m_2^\delta \mod p$. $A$ can be authenticated if and only if $\delta = \beta_{ik} - x_{ik}\omega \mod (p-1)$. From above, we have

$$Pr[S9] = \frac{1}{p-1} + \sum_{i=2}^{q_{active}} \left(\frac{1}{p-1} \prod_{j=2}^{i} \left(1 - \frac{1}{p} - j + 1\right)\right).$$

We can get the lower and upper bounds of $Pr[S9]$ as follows:

$$\frac{1}{p-1} + \frac{(p - q_{active} + 1)(1 - (1 - \frac{1}{p-q_{active}+1})^{q_{active} - 1})}{p-2} \leq Pr[S9],$$

$$Pr[S9] \leq \frac{1}{p-1} + \frac{(p - 1)(1 - \frac{1}{p})^{q_{active} - 1}}{p-q_{active}}.$$
Because $p$ is a large prime, the lower and upper bounds of $Pr[S9]$ are as follows:

$$\frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p - q_{\text{active}} + 1}\right)^{q_{\text{active}}^1}\right)$$

$$\leq Pr[S9] \leq \frac{1}{p-1} + \left(1 - \left(1 - \frac{1}{p - 1}\right)^{q_{\text{active}}^1}\right).$$

\[\Box \quad (11)\]

**Theorem 3.** Suppose solving one-way hash function and DLP are hard, no adversary $A$ can impersonate one party to cheat another legitimate party in authorization verification item adding and updating phase and cloud computing service accessing phase.

**Proof.** By Lemmas 8–11, upper bounds for probabilities of impersonating one party and being authenticated successfully are given. If solving one-way hash function and DLP are hard, the probabilities $\frac{1}{2^{\text{Len}h}}$ and $\frac{1}{p}$ are negligible such that probabilities of successful impersonation attacks approximate 0. For any adversary $A$, the probability to impersonate one party to cheat another legitimate party in authorization verification item adding and updating phase and cloud computing service accessing phase is negligible. \[\Box\]

### 4.2. Further Discussions

In the following, further discussions are made to show properties of the proposed mechanism.

**Secret retrieval and impersonation attack resistance:** By Theorems 2 and 3, no one can retrieve unauthorized secrets used for authentication, and no one can impersonate one party to cheat another. Thus, the proposed mechanism can resist secret retrieval and impersonation attack.

**Perfect forward secrecy:** In cloud computing service accessing phase, only legal $U$ and $S_j$ can authenticate each other and negotiate the shared key $m_8 = m_10 = g^{\sigma \times x \times s_{ik}} \mod p$. When other legal user $U'$ of $G_i$ knows $\{A_{ik}, \beta_{ik}, y_{ik}\}$ and tries to get the shared session key $g^{\sigma \times x \times s_{ik}} \mod p$, $U'$ can compute $s_{ik} = h(x_i || A_{ik} || \beta_{ik})$, $m_{1k}^1 \mod p = g^{\sigma \times s_{ik}} \mod p$, and $m_{2k}^2 \mod p = g^{\sigma \times x \times s_{ik}} \mod p$. However, $U'$ cannot obtain $g^{\sigma \times x \times s_{ik}} \mod p$ successfully. It is because DLP is hard to solve such that $U'$ cannot retrieve $\sigma / \alpha$ from $m_1 / m_2$. Even if the secret $\beta_{ik}$ is known, no one can retrieve previous session keys. Thus, the proposed mechanism provides perfect forward secrecy.

**Secret protection:** By Theorem 2, no one can get unauthorized secrets. Thus, the proposed mechanism ensures secret protection.

**Flexibility:** In the proposed mechanism, a trusted third party $R$ helps $G_i$ to register at $S_j$. $G_i$ can authorize $U$ to access specific cloud computing services with the given authorization item. $S_j$ uses the corresponding authorization verification item to verify whether $U$ is authorized by $G_i$. $G_i$ can generate multiple authorization items as needed and issue them to users appropriately. This approach provides high flexibility on authorization.

**Extensibility:** In the proposed mechanism, $S_j$ only authenticates the other party with the known secrets without managing authorization issues. $G_i$ can apply advanced cryptographic schemes to the proposed mechanism, such as encrypting the essential data with
a pre-determined secret key, without $G_i$’s help. This approach makes the essential information concealed even if it is kept by $S_j$. Thus, the proposed mechanism provides extensibility.

Audit: When authorized users possess the same rights to modify one specific document, the document owner or the system can trace who modifies it in the designed mechanism with a proper user authentication protocol (Chang et al., 2013; Liao et al., 2010). For audit, each user first registers at the system and needs to be authenticated to login to the system. After successfully authenticated, the user can perform operations on authorized resources or documents as in the original designed mechanism.

5. Conclusions

In this paper, an access control mechanism of verifiable cloud computing services is proposed by using chameleon hashing and Diffie–Hellman key exchange protocol. By the designed mechanism, an entity can apply for cloud computing services and he can authorize other users to access granted data or services. When an authorized user or entity wants to access cloud computing services, he can authenticate the cloud computing service provider. Moreover, no entity secret will be revealed by data kept by cloud servers such that security and cost saving can be both ensured. By the given security analyses, the security of the proposed mechanism is ensured. Moreover, the proposed mechanism provides secret retrieval and impersonation attack resistance, perfect forward secrecy, secret protection, flexibility, and extensibility. The proposed mechanism provides a novel and flexible solution to access control over verifiable cloud computing services.

Acknowledgements. This research was supported in part under grant No. NSC 101-2410-H-025-009-MY2 from the Ministry of Science and Technology, Taiwan.

References

Mathematics, 73, 463–468.
USA, pp. 143–154.
Y.-F. Chang is a Professor of Department of Computer Science and Information Engineering at National Taichung University of Science and Technology in Taiwan. She received her BSc degree in computer science and information engineering from National Chiao Tung University and PhD degree in computer science and information engineering from National Chung Cheng University, Taiwan. Her current research interests include electronic commerce, information security, cryptography, mobile communications, image processing, and data hiding.

Lanksti verifikuojamų debesų kompiuterijos paslaugų prieigos kontrolė su įrodumu saugumu

Ya-Fen CHANG