IBC-Based Entity Authentication Protocols for Federated Cloud Systems

Chenlei Cao[1], Ru Zhang[1, 2], Mengyi Zhang[1] and Yixian Yang[1]
[1] Information Security Center, and National Engineering Laboratory for Disaster Backup and Recovery,
Beijing University of Posts and Telecommunications
Beijing, 100876 – P.R.China
[e-mail: caochenlei@gmail.com, myzhang1988@gmail.com, yxyang@bupt.edu.cn]
[2] Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications
Beijing, 100876 – P.R.China
[e-mail: zhangru@bupt.edu.cn]
*Corresponding author: Chenlei Cao

Received September 28, 2013; revised March 26, 2013; accepted March 28, 2013; published May 31, 2013

Abstract

Cloud computing changes the service models of information systems and accelerates the pace of technological innovation of consumer electronics. However, it also brings new security issues. As one of the important foundations of various cloud security solutions, entity authentication is attracting increasing interest of many researchers. This article proposes a layered security architecture to provide a trust transmission mechanism among cloud systems maintained by different organizations. Based on the security architecture, four protocols are proposed to implement mutual authentication, data sharing and secure data transmission in federated cloud systems. The protocols not only can ensure the confidentiality of the data transferred, but also resist man-in-the-middle attacks and masquerading attacks. Additionally, the security properties of the four protocols have been proved by S-pi calculus formal verification. Finally, the performance of the protocols is investigated in a lab environment and the feasibility of the security architecture has been verified under a hybrid cloud system.

Keywords: Cloud Computing, Layered Security Architecture, Security Protocol, S-pi Calculus, Identity-Based Cryptography.

This work was supported by Beijing Natural Science Foundation (4122053), National Natural Foundation of China (61003284, 61121061), and The Major Projects of Digital Copyright Protection Technology Research and Development (GXTC-CZ-1015004/09, GXTC-CZ-1015004/15-1).

http://dx.doi.org/10.3837/tiis.2013.05.020

1. Introduction

As an emerging and powerful computational model, cloud computing enables ubiquitous, convenient, and on-demand network access to a shared pool of configurable computing resources. These resources can be quickly configured and released with minimal management cost and service provider interaction [1]. In a cloud system, a participant with the ability to process, send or receive information is referred to as a *cloud entity*, which can be a hardware device or a software process.

Compared to distributed computing and grid computing, cloud computing has more technical advantages [2], such as reliable system architecture, redundant data storage structure and centralized security policy management. However, due to the characteristics of cloud computing (for example, the delegated data management beyond the owner's control, widely distributed resource access nodes, flexible security policy and vague security boundary), the security problems of a cloud system are complex [2, 3]. It mainly involves the following aspects: system availability and reliability, data integrity and confidentiality, user privacy protection, audition ability, and security isolation among users. Adequate security policies need to be adopted to deal with these security problems. Among various security policies, effective authentication of entities is our major focus in this paper.

In recent years, many countries, companies and research groups have launched their own cloud systems, most of which are managed by a single cloud provider. However, there is a limitation for a single-provider cloud to cooperate with other cloud systems to maximize the function of available resources [24]. In order to achieve larger-scale resource sharing, to optimize resource allocation and to provide a seemingly infinite computing utility, Rochwerger et al. proposed the concept of federated clouds in the Reservoir project [25]. Although the new concept shows some attractive characteristics of the federated clouds for us, it brings some technical challenges [26]: first, how to establish standardized rules of hierarchical resource mapping in the process of resource delegation; second, how to design the strategies and the protocols for resource interaction among different resource delegations. As for the second type of problems, we studied many implementations of federated cloud systems and put forward our research point: when the user requests for resource integration, how the federated cloud system establishes secure data transmission channels among requesters, owners and delegations.

This article makes the following contributions to the field of authentication mechanism in a federated cloud system. Firstly, according to identity based cryptography (IBC) scheme [6], we propose a layered security architecture, which provides a secure trust transmission mechanism among cloud systems maintained by different organizations. Secondly, on the basis of the architecture, we present four security protocols to implement mutual authentication, data sharing and secure data transfer in the cloud. Thirdly, based on S-pi calculus [7, 8, 9], we prove the security properties of the protocols by means of formal verification. Finally, we implement the architecture and protocols, and investigate their performance by evaluating the Average Execution Times (AET) of the protocols. The test results show that the longest execution time of the protocol is shorter than 2.4 seconds in the circumstances of 100 concurrent requests.

The rest of the article is organized as follows: The related work is discussed in Section 2. The layered security architecture and the definitions are presented in Section 3. In Section 4, we propose the mutual authentication and data sharing protocols for federated cloud system.

Section 5 discusses the security properties of the protocols. Section 6 shows the experimental results of the security protocols, and Section 7 gives conclusions. The notations and S-pi calculus are provided in Appendix.

2. Related Work

As the widely used authentication protocols in the cloud system, Kerberos V5 [27] and OpenID V2.0 [29] still have some security vulnerabilities: Kerberos V5 cannot resist password-guessing attacks due to the poor passwords set by network users [27, 28]; Although timestamps in Kerberos V5 are supposed to prevent replay attacks, the protocol is still not robust enough, because the security of timestamps relies on the network time protocols and the ticket lifetimes. If the ticket lifetimes are too long, the adversary has sufficient time to replay old tickets and impersonate valid users [27]. OpenID V2.0 is vulnerable to phishing, identity-provider masquerade and denial-of-service attacks [29, 30]. Among various new solutions of authentication, asymmetric key authentication with a trusted third party has become a hotspot in recent years. In 2011, Grzonkowski and Corcoran [4, 5] proposed an authentication protocol based on the asymmetric key and zero-knowledge proof techniques to establish a secure resource sharing mechanism between home networks and the cloud system. Their work does not address the authentication problem in the federated cloud system; however, the zero-knowledge proof technique is very inspiring and promising to provide a technical means for future researches in the federated cloud system. In this paper, we propose a strong mutual authentication scheme to solve the problem of authentication for cloud entities in the federated cloud system.

2.1 Cryptography Basis

The most common system using trusted third-party for identity authentication is Public Key Infrastructure (PKI), where Certificate Authority (CA) serves as the trusted third-party to generate the public key certificate for the entity. However, the authentication process in the PKI scheme is very complicated, especially when entities managed by different CAs perform mutual authentication. Furthermore, if the entity cannot establish communication with CA, the mutual authentication mechanism becomes completely ineffective. In order to solve these problems, the IBC scheme [6] has been widely used. In that scheme, Private Key Generator (PKG) serves as the trusted third party to generate public key and the corresponding private key. Since the entity's identity is the public key, IBC is more suitable for cloud than PKI.

However, in the IBC scheme it is difficult for a single PKG to handle all authentication and authorization tasks in a large network. In order to alleviate the workload of PKG, Gentry et al. [13, 19] proposed a practical HIBE scheme to implement hierarchical entity registration and private key generation. Based on the decisional bilinear Diffie-Hellman assumption, Boneh and Boyen constructed another HIBE system [11, 14] in 2004 and refined it in 2011. The system has been proved to be secure in the selective-identity sense without random oracles. In 2005, Boneh and Boyen proposed an improved HIBE scheme [12], where the cipher text size of the scheme is independent of the hierarchy depth.

In the HIBE scheme, root PKG is responsible to generate private keys for lower-level PKGs, which in turn generate private keys for the entities in their own domains. When entity A intends to communicate with entity B, A only needs to obtain the identity-vector of B and the public parameters of the root PKG which B belongs to. Based on the Gentry's HIBE scheme, Lim et al. [15] proposed an identity-based key infrastructure for grid computing and designed

a transport layer security (TLS) handshake protocol. These fruitful researches provide good solutions for security problems in the cloud computing.

2.2 IBC-Based Authentication for Cloud Entity

In the field of cloud computing, Cheng et al. [16] proposed a data access scheme based on biometric authentication and IBE algorithm, but the application scope of the scheme is restricted by biometric algorithm accuracy, sensor stability, biometric privacy leak and masquerade. Schridde et al. [21, 22] employed only one PKG to provide service for all the entities in the cloud, and the security property of their system has not been fully proved. Kang and Zhang [18] proposed an IBC-based authentication scheme for the cloud storage model given by Kamara and Lauter [23]. In that scheme, child trusted nodes neither effectively share the load of root trusted node, nor participate in local trust management, and the scheme is inapplicable due to redundant structure and insecure key distribution. Based on HIBE scheme, Li et al. [17] proposed a hierarchical architecture for cloud computing and a TLS-analogous authentication protocol, but without mutual authentication scheme between service providers or between cloud systems. Yan et al. [19] adopted federated identity together with the HIBE scheme to implement federated identity management, key management and authentication among different types of clouds: public, private and hybrid clouds. In 2010, Huang et al. [20] gave another HIBE-based cloud system architecture, which realized a redundant backup for lower leveled PKGs in the cloud system.

Although the HIBE scheme provides an effective trust mechanism, it is much easier for the parent PKG to obtain the private information of the child PKG since the public parameter and the private key of the child PKG are all generated by the parent PKG. When cloud owners are in a competitive situation, it is difficult to establish a unified root PKG in a federated cloud system to provide trust service for different clouds. We propose a new layered cloud security architecture to establish a trust mechanism by employing global PKG in the federated cloud system, without leaking the information protected by the lower-level PKG.

3. System Architecture

3.1 Security Architecture

The layered cloud security architecture is illustrated in **Fig. 1**. We divide a federated cloud system into multiple cloud units. Each unit is a self-contained domain with an independent PKG responsible for certificating cloud entities in its own domain. We also set up a Global PKG (GPKG) as the global trusted authority to authenticate PKGs in different domains.

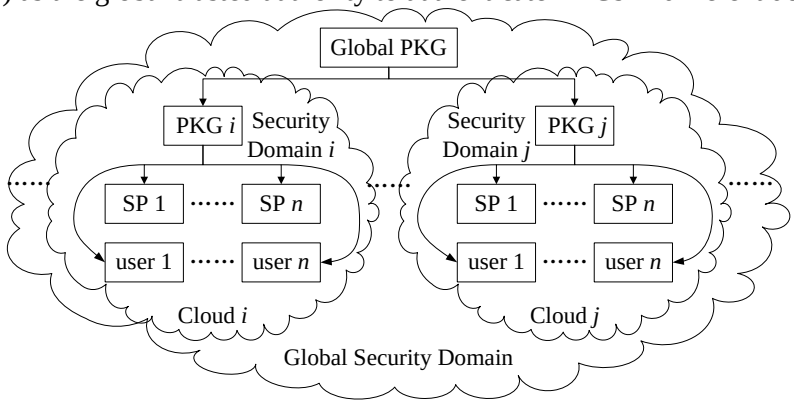

Fig. 1. Layered security architecture of the federated cloud system

For the security architecture depicted in **Fig. 1**, we give the following definitions:

Definition 1. Security Domain (SD). Divide the federated cloud system into multiple cloud units, and each unit is called a *security domain*.

Definition 2. Home Security Domain (HSD). The security domain in which the cloud entity registered is called the *home security domain* of the entity.

Definition 3. Routing Security Domain (RSD). The non-home security domain accessed by the cloud entity in another security domain is called the *routing security domain*.

Based on this architecture, we propose a set of mutual authentication and data sharing protocols for the federated cloud system in Section 4.

3.2 Cloud Entity Authorization

The process of cloud entity authorization includes two parts: GPKG authorizes PKG in each SD; PKG authorizes cloud entities, such as users and service providers (SPs) in its own SD. Before accessing the cloud system, the cloud entity must register its identifiable information in a SD. The corresponding PKG generates an authentication vector and binds it to the entity.

3.2.1 PKG Registration and Authorization

Step 1. PKG registers its identifiable information at GPKG. GPKG authenticates the identifiable information and generates a public key $ID_{PKG} = (id_{PKG}, T)$ for the PKG, where T represents the valid period of ID_{PKG} and id_{PKG} the PKG's identifiable information.

Step 2. Based on the BF-IBE scheme [6], GPKG computes $G(K_{GKPG}, P_{GPKG}, ID_{PKG})$ to generate the private key K^{-}_{PKG} corresponding to ID_{PKG}, where K_{GPKG} is the GPKG's master key and P_{GPKG} the public parameter generated by GPKG.

Step 3. GPKG generates an authentication vector $V_{RPKG} = (K^{-}_{RPKG}, P_{GPKG}, ID_{GPKG}, RAI)$ and binds it to the PKG via a secure channel (either based on smartcards or security protocols), where ID_{GPKG} is the GPKG's identity and RAI the register area identifier allocated by GPKG.

3.2.2 User Authorization

Step 1. The user registers his identifiable information id_{user} at PKG in arbitrary SD and gets a public key $ID_{user} = (id_{user}, T)$, where T is the valid period of ID_{user}.

Step 2. The PKG generates the user's private key K^{-}_{user} by $G(K_{KPG}, P_{PKG}, ID_{user})$, where K_{PKG} is the PKG's master key and P_{PKG} is the public parameter.

Step 3. The PKG generates an authentication vector $V_{user} = (K^{-}_{user}, P_{PKG}, ID_{PKG}, RAI)$ and binds it to the user via a secure channel, where ID_{PKG} is the public key of PKG.

3.2.3 SP Authorization

Similarly, SP registers its identifiable information at PKG in arbitrary SD and gets a public key $ID_{SP} = (id_{SP}, T)$. PKG generates a vector $V_{SP} = (K^{-}_{SP}, P_{PKG}, ID_{PKG}, RAI)$ and binds it to the SP, where K^{-}_{SP} is the SP's private key and id_{SP} the SP's identifiable information.

3.3 Key Update Mechanism

Based on the authorization mechanism above, each cloud entity can obtain a set of public keys in the form of $\{(id, T_1), (id, T_2), \cdots (id, T_n)\}$, where T_i is a time series code marking the valid period of each public key and id represents the identifiable information of each entity. If the public key (id, T_i) expires, the entity will enable a new one (id, T_{i+1}) and its corresponding

private key changes automatically. In this way, the cloud system can manage the key pairs pre-bound to the cloud entity by T_i.

4. Security Protocols

4.1 Mutual Authentication Protocol between PKGs in Global Security Domain

In order to establish an effective trust transmission mechanism between different SDs, PKGs need to authenticate each other and exchange their public parameters. The mutual authentication protocol is illustrated in **Fig. 2**.

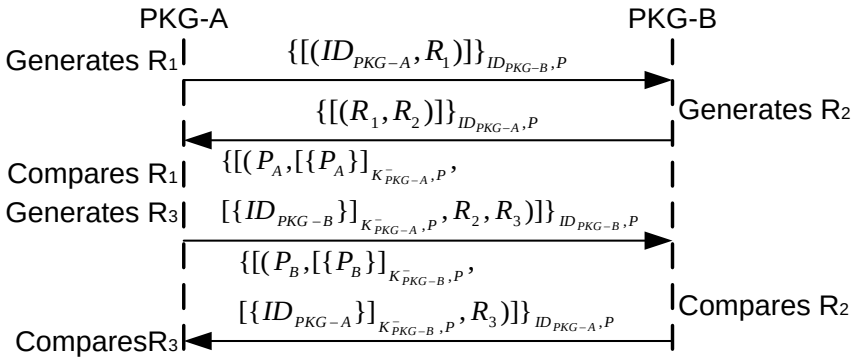

Fig. 2. Mutual authentication protocol between PKGs

1. PKG-A generates a random number R_1 and encrypts (ID_{PKG-A}, R_1) with PKG-B's public key ID_{PKG-B}. PKG-A sends a request vector to PKG-B: $\{[(ID_{PKG-A}, R_1)]\}_{ID_{PKG-B},P}$, where P is the public parameter generated by GPKG.

2. On receiving PKG-A's request vector, PKG-B decrypts $\{[(ID_{PKG-A}, R_1)]\}_{ID_{PKG-B},P}$ with its private key K^-_{PKG-B} to retrieve R_1 and ID_{PKG-A}, and then generates a random number R_2. PKG-B sends a response vector to PKG-A: $\{[(R_1, R_2)]\}_{ID_{PKG-A},P}$.

3. Upon the receipt of PKG-B's response vector, PKG-A decrypts it with its private key K^-_{PKG-A} to get R_1 and R_2, and then compares R_1 with the random number PKG-A has generated before to authenticate the identity of PKG-B. If the authentication succeeds, PKG-A generates a random number R_3, signatures $[\{P_A\}]_{K^-_{PKG-A},P}$ and $[\{ID_{PKG-A}\}]_{K^-_{PKG-A},P}$, and sends them to PKG-B: $\{[(P_A, [\{P_A\}]_{K^-_{PKG-A},P}, [\{ID_{PKG-B}\}]_{K^-_{PKG-A},P}, R_2, R_3)]\}_{ID_{PKG-B},P}$, where P_A is the public parameter of PKG-A.

4. As soon as PKG-B receives the response vector from PKG-A, PKG-B decrypts it to retrieve P_A, R_2, R_3 and $[\{ID_{PKG-B}\}]_{K^-_{PKG-A},P}$, and then compares R_2 with the random number PKG-B has generated before to verify the authenticity of PKG-A. If they match, PKG-B verifies the validity of P_A through $[\{P_A\}]_{K^-_{PKG-A},P}$. If the verification succeeds, PKG-B generates a signature $[\{P_B\}]_{K^-_{PKG-B},P}$ and sends a response vector to PKG-A:

$$\{[(P_B, [\{P_B\}]_{K^-_{PKG-B},P}, [\{ID_{PKG-A}\}]_{K^-_{PKG-B},P}, R_3)]\}_{ID_{PKG-A},P},$$

where P_B is the public parameter of PKG-B.

5. On receiving PKG-B's response vector, PKG-A decrypts it to retrieve P_B, R_3 and $[\{ID_{PKG-A}\}]_{K^-_{PKG-B},P}$, and then compares R_3 with the random number PKG-A has generated

before to verify the timeliness of the vector. After that, PKG-A verifies the authenticity of P_B by checking $[\{P_B\}]_{K^-_{PKG-B},P}$ to complete the exchanging process of public parameters.

Assume an instance that PKG i (as an initiator) and PKG j (as a responder) in different SDs need to exchange their public parameters. Based on S-pi calculus, the process functions corresponding to the initiator, the responder and the whole protocol are expressed as follows:

Definition 4. The process function of the initiator.

$$\begin{aligned} &P_{PKG}(i,j) \triangleq \overline{C_{PKGj}}\langle\{[(ID_{PKGi}, R_{i1,j})]\}_{ID_{PKGj},P}\rangle.C_{PKGi}(x_{i1,j}).case\ x_{i1,j}\ of\ \{[x_{i2,j}]\}_{K^-_{PKGi},P}\ in \\ &let(x_{i21,j}, x_{i22,j}) = x_{i2,j}\ in [x_{i21,j}\ is\ R_{i1,j}]\overline{C_{PKGj}}\langle\{[(P_i,[\{P_i\}]_{K^-_{PKGi},P},[\{ID_{PKGj}\}]_{K^-_{PKGi},P}, x_{i22,j}, R_{i3,j})]\}_{ID_{PKGj},P}\rangle. \\ &C_{PKGi}(x_{i3,j}).case\ x_{i3,j}\ of\ \{[x_{i4,j}]\}_{K^-_{PKGi},P}\ in\ let(x_{i41,j}, x_{i42,j}, x_{i43,j}, x_{i44,j}) = x_{i4,j}\ in \\ &[x_{i44,j}\ is\ R_{i3,j}]\ case\ x_{i42,j}\ of\ [\{x_{i41,j}\}]_{ID_{PKGj},P}\ in\ F(x_{i41,j}, x_{i42,j}, x_{i43,j}) \end{aligned} \quad (1)$$

Definition 5. The process function of the responder.

$$\begin{aligned} &P_{PKG'}(i,j) \triangleq C_{PKGj}(y_{i,j1}).case\ y_{i,j1}\ of\ \{[y_{i,j2}]\}_{K^-_{PKGj},P}\ in\ let(y_{i,j21}, y_{i,j22}) = y_{i,j2}\ in \\ &\overline{C_{PKGi}}\langle\{[(y_{i,j22}, R_{i,j2})]\}_{y_{i,j21},P}\rangle.C_{PKGj}(y_{i,j3}).case\ y_{i,j3}\ of\ \{[y_{i,j4}]\}_{K^-_{PKGj},P}\ in \\ &let(y_{i,j41}, y_{i,j42}, y_{i,j43}, y_{i,j44}, y_{i,j45}) = y_{i,j4}\ in [y_{i,j44}\ is\ R_{i,j2}]\ case\ y_{i,j42}\ of\ [\{y_{i,j41}\}]_{y_{i,j21},P} \\ &in\ \overline{C_{PKGi}}\langle\{[(P_j,[\{P_j\}]_{K^-_{PKGj},P},[\{y_{i,j21}\}]_{K^-_{PKGj},P}, y_{i,j45})]\}_{y_{i,j21},P}\rangle \end{aligned} \quad (2)$$

Definition 6. The whole protocol function.

$$\begin{aligned} &Sys(I_1, I_2, \cdots I_n) \triangleq \prod_{i\in N}(vK^-_{PKGi})\prod_{j\in N}(vK^-_{PKGj})(\prod_{n\in N}(P_{PKG}(I_n) \mid P_{PKG'}(I_n))) \\ &I_n = (i,j), \forall n \neq m \Leftrightarrow I_n \neq I_m \end{aligned} \quad (3)$$

4.2 User Access Authentication Protocol in Routing Security Domain

When a user belonging to security domain A needs to access SP in security domain B, the access authentication protocol is illustrated in **Fig. 3**.

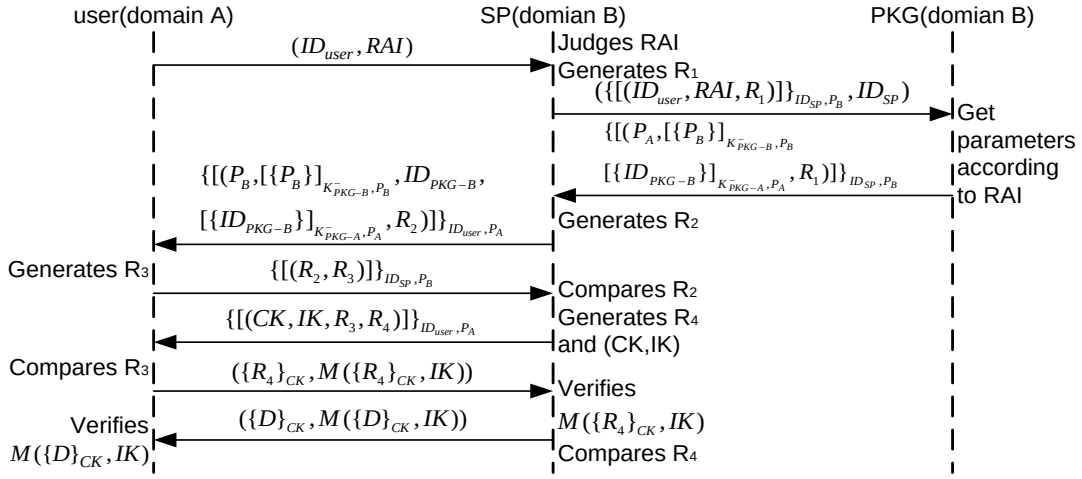

Fig. 3. User access authentication protocol in RSD

1. The user in security domain A sends SP an access request vector: (ID_{user}, RAI).

2. Upon the receipt of the request vector, SP in security domain B judges the user's HSD through RAI. After that, SP generates a random number R_1 and sends a request vector to

PKG-B to apply for the public parameter of security domain A and the signature of the public parameter of security domain B. SP sends a vector to PKG-B: $(\{[(ID_{user}, RAI, R_1)]\}_{ID_{SP},P_B}, ID_{SP})$.

3. On receiving SP's request vector, PKG-B decrypts it with K_{SP}^{-} to retrieve R_1, ID_{user} and RAI, and then judges the user's HSD through RAI. After that, according to the protocol in Section 4.1, PKG-B communicates with PKG-A to get the public parameters of security domain A and their signatures, and then PKG-B sends a response vector to SP: $\{[(P_A,[\{P_B\}]_{K_{PKG-B}^{-},P_B}[\{ID_{PKG-B}\}]_{K_{PKG-A}^{-},P_A}, R_1)]\}_{ID_{SP},P_B}$.

4. On receiving the response vector from PKG-B, SP decrypts it with K_{SP}^{-} to retrieve R_1, and then compares it with the random number SP has generated before to verify the timeliness of the response vector. If the verification succeeds, SP generates a random number R_2 and sends a response vector to the user: $\{[(P_B,[\{P_B\}]_{K_{PKG-B}^{-},P_B}, ID_{PKG-B},[\{ID_{PKG-B}\}]_{K_{PKG-A}^{-},P_A}, R_2)]\}_{ID_{user},P_A}$.

5. Upon the receipt of the response vector, the user decrypts it with K_{user}^{-} and verifies $[\{ID_{PKG-B}\}]_{K_{PKG-A}^{-},P_A}$ with ID_{PKG-A} to confirm the authenticity of ID_{PKG-B}. If the verification succeeds, the user verifies $[\{P_B\}]_{K_{PKG-B}^{-},P_B}$ with ID_{PKG-B} to confirm the authenticity of P_B. If the verification succeeds, the user generates a random number R_3 and sends a response vector to SP: $\{[(R_2, R_3)]\}_{ID_{SP},P_B}$.

6. Upon the receipt of the response vector from the user, SP decrypts it to retrieve (R_2, R_3), and compares R_2 with the random number SP has generated before to verify the user's identity. If they match, SP generates a random number R_4 and a session key vector (CK, IK), where CK is the cipher key and IK the integrity key. SP sends a response vector to the user:

$$\{[(CK, IK, R_3, R_4)]\}_{ID_{user},P_A}.$$

7. On receiving the response vector, the user decrypts it to retrieve R_3, R_4 and (CK, IK). Then the user verifies the timeliness of (CK, IK) through R_3. If the verification succeeds, the user sends a response vector to SP: $(\{R_4\}_{CK}, M(\{R_4\}_{CK}, IK))$.

8. Upon the receipt of the user's response vector, SP verifies the integrity of the vector through $M(\{R_4\}_{CK}, IK)$. If the verification succeeds, SP decrypts $\{R_4\}_{CK}$ with CK to retrieve R_4 and compares it with the random number SP has generated before to confirm that the user has received the session keys correctly. If they match, SP encrypts D with (CK, IK) and sends the results of encryption to the user: $(\{D\}_{CK}, M(\{D\}_{CK}, IK))$.

Assume an instance that user i-A belonging to security domain A needs to access SP j-B in security domain B. Based on S-pi calculus, the process functions corresponding to user i-A, SP j-B, PKG-B and the whole protocol are expressed as follows:

Definition 7. The user's process function for the access protocol in RSD.

$$\begin{aligned}
&P_{user-A}(i,j) \triangleq \overline{C_{SPj-B}}\langle (ID_{useri-A}, RAI)\rangle_{\bullet} C_{useri-A}(x_{i1,j})_{\bullet} case\ x_{i1,j}\ of\ \{[x_{i2,j}]\}_{K_{useri-A}^{-},P_A}\ in\\
&let(x_{i21,j}, x_{i22,j}, x_{i23,j}, x_{i24,j}, x_{i25,j}) = x_{i2,j}\ in\ case\ x_{i24,j}\ of\ [\{x_{i23,j}\}]_{ID_{PKG-A},P_A}\ in\ case\ x_{i22,j}\ of\\
&[\{x_{i21,j}\}]_{x_{i23,j},x_{i21,j}}\ in\ \overline{C_{SPj-B}}\langle \{[(x_{i25,j}, R_{i3,j})]\}_{ID_{SPj-B},x_{i21,j}}\rangle_{\bullet} C_{useri-A}(x_{i3,j})_{\bullet} case\ x_{i3,j}\ of\\
&[\{x_{i4,j}\}]_{K_{useri-A}^{-},P_A}\ in\ let(x_{i41,j}, x_{i42,j}, x_{i43,j}, x_{i44,j}) = x_{i4,j}\ in\ [x_{i43,j}\ is\ R_{i3,j}]\\
&\overline{C_{SPj-B}}\langle (\{x_{i44,j}\}_{x_{i41,j}}, M(\{x_{i44,j}\}_{x_{i41,j}}, x_{i42,j}))\rangle_{\bullet} C_{useri-A}(x_{i5,j})_{\bullet} let(x_{i51,j}, x_{i52,j}) = x_{i5,j}\ in\\
&[M(x_{i51,j}, x_{i42,j})\ is\ x_{i52,j}]\ case\ x_{i51,j}\ of\ \{x_{i6,j}\}_{x_{i41,j}}\ in\ F(x_{i6,j})
\end{aligned} \tag{4}$$

Definition 8. The SP's process function for the access protocol in RSD.

$$
\begin{aligned}
&P_{SP-B}(i,j,D) \stackrel{\Delta}{=} (vIK_{i,j})(vCK_{i,j})C_{SPj-B}(y_{i,j1}).let(y_{i,j11},y_{i,j12}) = y_{i,j1}\ in[y_{i,j12}\ is\ not\ RAI_B] \\
&\overline{C_{PKG-B}}\left\langle(\{[(y_{i,j1},R_{i,j1})]\}_{ID_{SPj-B},P_B},ID_{SPj})\right\rangle.C_{SPj-B}(y_{i,j2}).case\ y_{i,j2}\ of\ \{[y_{i,j3}]\}_{K^-_{SPj-B},P_B}\ in \\
&let(y_{i,j31},y_{i,j32},y_{i,j33},y_{i,j34}) = y_{i,j3}\ in[y_{i,j34}\ is\ R_{i,j1}]\overline{C_{useri-A}}\left\langle\begin{matrix}\{[(P_B,y_{i,j32},ID_{PKG-B},\\ y_{i,j33},R_{i,j2})]\}_{y_{i,j11},y_{i,j31}}\end{matrix}\right\rangle. \\
&C_{SPj-B}(y_{i,j4}).case\ y_{i,j4}\ of\ \{[y_{i,j5}]\}_{K^-_{SPj-B},P_B}\ in\ let(y_{i,j51},y_{i,j52}) = y_{i,j5}\ in[y_{i,j51}\ is\ R_{i,j2}] \\
&\overline{C_{useri-A}}\left\langle\{[(CK_{i,j},IK_{i,j},y_{i,j52},R_{i,j4})]\}_{y_{i,j11},y_{i,j31}}\right\rangle.C_{SPj-B}(y_{i,j6}).let(y_{i,j61},y_{i,j62}) = y_{i,j6}\ in \\
&[M(y_{i,j61},IK_{i,j})\ is\ y_{i,j62}]\ case\ y_{i,j61}\ of\ \{y_{i,j7}\}_{CK_{i,j}}\ in\ [y_{i,j7}\ is\ R_{i,j4}] \\
&\overline{C_{useri-A}}\left\langle(\{D\}_{CK_{i,j}},M(\{D\}_{CK_{i,j}},IK_{i,j}))\right\rangle
\end{aligned}
\tag{5}
$$

Definition 9. The PKG-B's process function for the access protocol in RSD.

$$
\begin{aligned}
&P_{PKG-B}(j) \stackrel{\Delta}{=} C_{PKG-B}(z_{j,k1}).let(z_{j,k11},z_{j,k12}) = z_{j,k1}\ in\ [z_{j,k12}\ is\ ID_{SPj-B}]\ case\ z_{j,k11}\ of\ \{[z_{j,k2}]\}_{K^-_{SPj-B},P_B} \\
&in\ let(z_{j,k21},z_{j,k22},z_{j,k23}) = z_{j,k2}\ in\ [z_{j,k22}\ is\ RAI_A]\overline{C_{SPj-B}}\left\langle\begin{matrix}\{[(P_A,[\{P_B\}]_{K^-_{PKG-B},P_B},\\ [\{ID_{PKG-B}\}]_{K^-_{PKG-A},P_A},z_{j,k23})]\}_{ID_{SPj-B},P_B}\end{matrix}\right\rangle
\end{aligned}
\tag{6}
$$

Definition 10. The whole system function for the access protocol in RSD.

$$
\begin{aligned}
&Sys(I_1,I_2,\cdots I_n) \stackrel{\Delta}{=} (vK^-_{PKG-B})\prod_{i\in N}(vK^-_{useri-A})\prod_{j\in N}(vK^-_{SPj-B}) \\
&(\prod_{n\in N}(P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n))),\quad I_n=(i,j),\forall n\neq m\Leftrightarrow I_n\neq I_m
\end{aligned}
\tag{7}
$$

4.3 Mutual Authentication Protocol between SPs in Different Security Domains

When SP-A in domain A needs to set up a secure communication channel with SP-B in domain B, the mutual authentication protocol is illustrated in **Fig. 4**. Since the principle of the protocol between SPs in different SDs is the same with the protocol in Section 4.2, the implementation details and the security analysis of the protocol are omitted in this article.

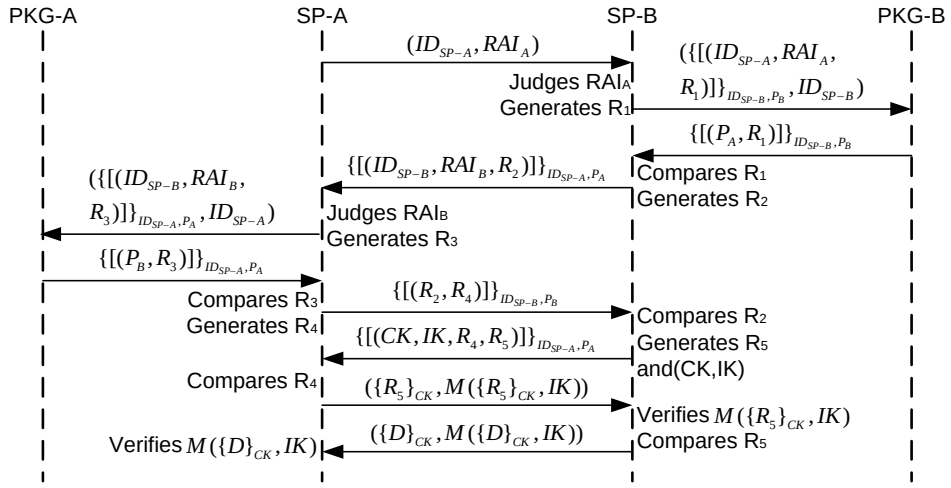

Fig. 4. Mutual authentication protocol between SPs in different SDs

4.4 Data Sharing Protocol in Different Security Domains

In the scenario where the user belonging to security domain A needs to share his data stored in SP-A with SP-B, the data sharing protocol between SPs is illustrated in **Fig. 5**, with the premise that the user has accessed the cloud system in RSD and SPs have established a secure data transmission channel. The detailed security analysis of this protocol is omitted in our article, because the principle of the protocol is the similar with the previous protocols.

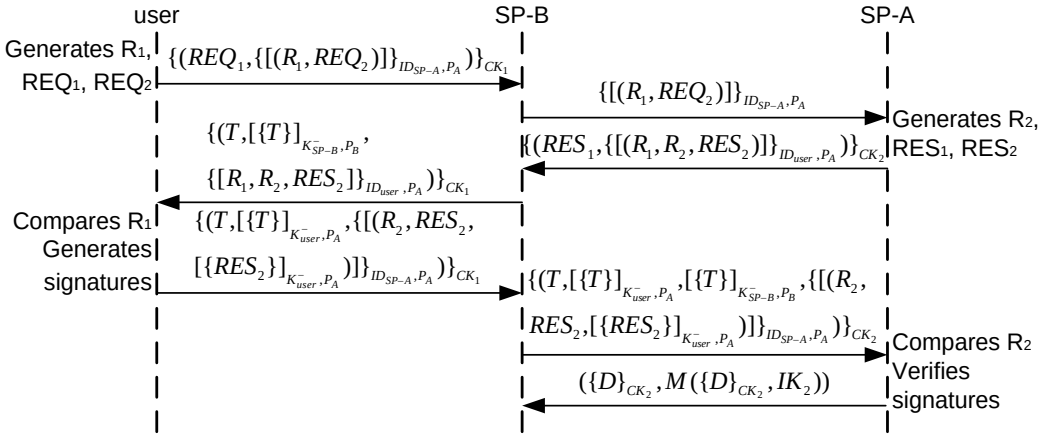

Fig. 5. Data sharing protocol in different SDs

1. The user sends a request vector to SP-B: $\{(REQ_1,\{[(R_1,REQ_2)]\}_{ID_{SP-A},P_A})\}_{CK_1}$. REQ_1 is the data-processing request sent to SP-B. REQ_2 is the data-sharing request sent to SP-A. R_1 is a random number. CK_1 is the session key between the user and SP-B.

2. Upon the receipt of the user's request vector, SP-B decrypts it with CK_1 to get the user's requirement through REQ_1. After that, SP-B sends $\{[(R_1,REQ_2)]\}_{ID_{SP-A},P_A}$ to SP-A.

3. On receiving the user's request forwarded by SP-B, SP-A decrypts it with K^-_{SP-A} to get the user's service requirement through REQ_2 and then sends a response vector to SP-B:

$$\{(RES_1,\{[(R_1,R_2,RES_2)]\}_{ID_{user},P_A})\}_{CK_2}.$$

RES_1 is a service-response vector sent to SP-B. RES_2 is a service-response vector sent to the user. R_2 is a random number. CK_2 is the session key between SP-B and SP-A.

4. As soon as SP-B receives the response vector from SP-A, SP-B decrypts it with CK_2. According to RES_1, the SP-B generates a service token T, and sends a request vector to the user asking for a service authorization: $\{(T,[\{T\}]_{K^-_{SP-B},P_B},\{[R_1,R_2,RES_2]\}_{ID_{user},P_A})\}_{CK_1}$, where $[\{T\}]_{K^-_{SP-B},P_B}$ is the signature of T signed by SP-B.

5. Upon the receipt of SP-B's response vector, the user decrypts it with CK_1 to retrieve $\{[R_1,R_2,RES_2]\}_{ID_{user},P_A}$, T and $[\{T\}]_{K^-_{SP-B},P_B}$. The user confirms the validity of T by checking the authenticity of $[\{T\}]_{K^-_{SP-B},P_B}$. If the confirmation succeeds, the user generates an authorization signature $[\{T\}]_{K^-_{user},P_A}$ and decrypts $\{[R_1,R_2,RES_2]\}_{ID_{user},P_A}$ to verify the validity of RES_2 through R_1. If the verification succeeds, the user gets SP-A's response through RES_2 and sends an authorization vector to SP-B: $\{(T,[\{T\}]_{K^-_{user},P_A},\{[(R_2,RES_2,[\{RES_2\}]_{K^-_{user},P_A})]\}_{ID_{SP-A},P_A})\}_{CK_1}$.

6. On receiving the user's response vector, SP-B decrypts it with CK_1 and validates the user's authorization signature. SP-B sends a response vector to SP-A:

$$\{(T,[\{T\}]_{K^{-}_{user},P_A},[\{T\}]_{K^{-}_{SP-B},P_B},\{[(R_2,RES_2,[\{RES_2\}]_{K^{-}_{user},P_A})]\}_{ID_{SP-A},P_A})\}_{CK_2}\ .$$

7. Upon the receipt of SP-B's response vector, SP-A decrypts it with CK_2 and verifies the validity of T through $[\{T\}]_{K^{-}_{user},P_A}$ and $[\{T\}]_{K^{-}_{SP-B},P_B}$. If the verification succeeds, SP-A decrypts $\{[(R_2,RES_2,[\{RES_2\}]_{K^{-}_{user},P_A})]\}_{ID_{SP-A},P_A}$ and verifies the timeliness of RES_2 through R_2 and the authenticity of RES_2 through $[\{RES_2\}]_{K^{-}_{user},P_A}$. If the validations succeed and the service request described in T is consistent with what is described in RES_2, SP-A shares the user's data with SP-B: $(\{D\}_{CK_2},M(\{D\}_{CK_2},IK_2))$.

After user i-A accesses SP j-B, he requests SP j-B to get his data stored in SP k-A. Based on S-pi calculus, the process functions corresponding to user i-A, SP j-B, SP k-A and the whole system are expressed as follows:

Definition 16. The user's process function for the data sharing protocol.

$$\begin{aligned}
&P_{user-A}(i,j,k)\overset{\Delta}{=}\overline{C_{SPj-B}}\left\langle\{(REQ_{i1,j},\{[(R_{i1,k},REQ_{i2,k})]\}_{ID_{SPk-A},P_A})\}_{CK_{i,j}}\right\rangle.C_{useri-A}(x_{i1,j}).\\
&case\ x_{i1,j}\ of\ \{x_{i2,j}\}_{CK_{i,j}}\ in\ let(x_{i21,j},x_{i22,j},x_{i23,j})=x_{i2,j}\ in\ case\ x_{i22,j}\ of\ [\{x_{i21,j}\}]_{ID_{SPj-B},P_B}\\
&in\ case\ x_{i23,j}\ of\ \{[x_{i3,j}]\}_{K^{-}_{useri-A},P_A}\ in\ let(x_{i31,j},x_{i32,j},x_{i33,j})=x_{i3,j}\ in\ [x_{i31,j}\ is\ R_{i1,k}]\\
&\overline{C_{SPj-B}}\left\langle\{(x_{i21,j},[\{x_{i21,j}\}]_{K^{-}_{useri-A},P_A},\{[x_{i32,j},x_{i33,j},[\{x_{i33,j}\}]_{K^{-}_{useri-A},P_A}]\}_{ID_{SPk-A},P_A})\}_{CK_{i,j}}\right\rangle
\end{aligned}\tag{8}$$

Definition 17. The receiver (SP j-B) process function for the data sharing protocol.

$$\begin{aligned}
&P_{SP-B}(i,j,k)\overset{\Delta}{=}C_{SPj-B}(y_{i,j1}).case\ y_{i,j1}\ of\ \{y_{i,j2}\}_{CK_{i,j}}\ in\ let(y_{i,j21},y_{i,j22})=y_{i,j2}\ in\\
&\overline{C_{SPk-A}}\left\langle y_{i,j22}\right\rangle.C_{SPj-B}(y_{j3,k}).case\ y_{j3,k}\ of\ \{y_{j4,k}\}_{CK_{j,k}}\ in\ let(y_{j41,k},y_{j42,k})=y_{j4,k}\ in\\
&\overline{C_{useri-A}}\left\langle\{(T,[\{T\}]_{K^{-}_{SPj-B},P_B},y_{j42,k})\}_{CK_{i,j}}\right\rangle.C_{SPj-B}(y_{i,j5}).case\ y_{i,j5}\ of\ \{y_{i,j6}\}_{CK_{i,j}}\ in\\
&let(y_{i,j61},y_{i,j62},y_{i,j63})=y_{i,j6}\ in\ [y_{i,j61}\ is\ T]case\ y_{i,j62}\ of\ [\{T\}]_{ID_{useri-A},P_A}\ in\\
&\overline{C_{SPk-A}}\left\langle\{(T,y_{i,j62},[\{T\}]_{K^{-}_{SPj-B},P_B},y_{i,j63})\}_{CK_{j,k}}\right\rangle.C_{SPj-B}(y_{j7,k}).let(y_{j71,k},y_{j72,k})=y_{j7,k}\ in\\
&[M(y_{j71,k},IK_{j,k})\ is\ y_{j72,k}]case\ y_{j71,k}\ of\ \{y_{j8,k}\}_{CK_{j,k}}\ in\ F(y_{j8,k})
\end{aligned}\tag{9}$$

Definition 18. The sender (SP k-A) process function for the data sharing protocol.

$$\begin{aligned}
&P_{SP-A}(i,j,k,D)\overset{\Delta}{=}C_{SPk-A}(z_{j,k1}).case\ z_{j,k1}\ of\ \{[z_{j,k2}]\}_{K^{-}_{SPk-A},P_A}\ in\ let(z_{j,k21},z_{j,k22})=z_{j,k2}\ in\\
&\overline{C_{SPj-B}}\left\langle\{(RES_{j,k1},\{[(z_{j,k21},R_{i,k2},RES_{i,k2})]\}_{ID_{useri-A},P_A})\}_{CK_{j,k}}\right\rangle.C_{SPk-A}(z_{j,k3}).case\ z_{j,k3}\ of\\
&\{[z_{j,k4}]\}_{K_{j,k}}\ in\ let(z_{j,k41},z_{j,k42},z_{j,k43},z_{j,k44})=z_{j,k4}\ in\ case\ z_{j,k42}\ of\ [\{z_{j,k41}\}]_{ID_{useri-A},P_A}\ in\\
&case\ z_{j,k43}\ of\ [\{z_{j,k41}\}]_{ID_{SPj-B},P_B}\ in\ case\ z_{j,k44}\ of\ \{[z_{j,k5}]\}_{K^{-}_{SPk-A},P_A}\ in\\
&let(z_{j,k51},z_{j,k52},z_{j,k53})=z_{j,k5}\ in\ [z_{j,k51}\ is\ R_{i,k2}][z_{j,k52}\ is\ RES_{i,k2}]case\ z_{j,k53}\ of\\
&[\{RES_{i,k2}\}]_{ID_{useri-A},P_A}\ in\ \overline{C_{SPj-B}}\left\langle(\{D\}_{CK_{j,k}},M(\{D\}_{CK_{j,k}},IK_{j,k}))\right\rangle
\end{aligned}\tag{10}$$

Definition 19. The whole system function for the data sharing protocol.

$$\begin{aligned}
&Sys(I_1,I_2,\cdots I_n)\overset{\Delta}{=}\prod_{i\in N}(vK^{-}_{useri-A})\prod_{j\in N}(vK^{-}_{SPj-B})\prod_{k\in N}(vK^{-}_{SPk-A})\\
&(\prod_{n\in N}(P_{user-A}(I_n)\mid P_{SP-B}(I_n)\mid P_{SP-A}(I_n,D_n))),\ I_k=(i,j,k),\forall n\neq m\Leftrightarrow I_n\neq I_m
\end{aligned}\tag{11}$$

5. Security Analysis Based on S-pi Calculus

Based on S-pi calculus, this section provides proofs for the security properties of the protocols in two areas: secrecy and authenticity.

5.1 Secrecy Analysis

Our security architecture includes two-level trust nodes: GPKG in the global security domain (GSD) and PKG in the lower-level security domain (LSD). In GSD, GPKG is in charge of setting up trust mechanism among PKGs in different SDs. In LSD, PKG is responsible for setting up trust mechanism among cloud entities registered in its domain.

5.1.1 Secrecy Analysis of the Protocol between PKGs in GSD

Following (1) and (2) in Section 4.1, let C_{PKGi} represent the communication channel of PKG i and C_{PKGj} represents the communication channel of PKG j. In order to prove the secrecy property of the protocol, we need to prove Proposition 1-3.

Proposition 1. The protocol process functions execute successfully only if C_{PKGi} and C_{PKGj} denote the same communication channel.

Proof. According to the structural equivalence axioms (36)-(42) in appendix, function (3) can be transformed as:

$$\begin{aligned} &Sys(I_1, I_2, \cdots I_n) \equiv \prod_{i\in N}(v\bar{K}_{PKGi})\prod_{j\in N}(v\bar{K}_{PKGj})(\prod_{n\in N}(P_{PKG}(I_n) \mid P_{PKG'}(I_n))) \equiv \\ &\prod_{i\in N}(v\bar{K}_{PKGi})\prod_{j\in N}(v\bar{K}_{PKGj})(P_{PKG}(I_1) \mid P_{PKG}(I_2)\cdots \mid P_{PKG}(I_n) \mid P_{PKG'}(I_1) \mid P_{PKG'}(I_2)\cdots \mid P_{PKG'}(I_n)) \end{aligned} \tag{12}$$

where $\{I_n = (i, j) \mid \forall n \neq m \Leftrightarrow I_n \neq I_m\}$.

Any two terms of $P_{PKG}(I_n)$ and $P_{PKG'}(I_m)$ in (12) can compose a group of protocol processes:

$$P_{PKG}(I_n) \mid P_{PKG}(I_m) \equiv P_{PKGi}(i, j) \mid P_{PKGj}(k, l) \tag{13}$$

where $\{I_n = (i, j), I_m = (k, l)\}$.

According to the protocol processes defined in (1) and (2), equation (13) can be further expressed as (14).

$$\begin{aligned} P_{PKGi}(I_n) \mid P_{PKGj}(I_m) \equiv{}& \overline{C_{PKGj}}\langle\sigma_1\rangle_{\bullet} C_{PKGi}(x_1)_{\bullet} N_1(x_1)\, \overline{C_{PKGj}}\langle\sigma_2\rangle_{\bullet} C_{PKGi}(x_2)_{\bullet} N_2(x_2) \mid \\ & C_{PKGl}(\theta_1)_{\bullet} M_1(\theta_1)\, \overline{C_{PKGk}}\langle y_1\rangle_{\bullet} C_{PKGl}(\theta_2)_{\bullet} M_2(\theta_2)\, \overline{C_{PKGk}}\langle y_2\rangle \end{aligned} \tag{14}$$

The data sent/received by PKG j / PKG i is represented by σ_i / x_i respectively. $N_i(x_i)_{i=1,2,3}$ is the function to process x_i. The data sent/ received by PKG k / PKG l is represented by y_j / θ_j respectively. $M_j(y_j)_{j=1,2,3}$ is the function to process y_j.

For $\{I_n = (i, j), I_m = (k, l) \mid I_n \neq I_m\}$, there are three cases: $i = k \cap j \neq l$, $i \neq k \cap j = l$ and $i \neq k \cap j \neq l$. If $i = k \cap j \neq l$ or $i \neq k \cap j \neq l$, we have $C_{PKGj} \neq C_{PKGl}$. The protocol process expressed by (14) cannot execute successfully, because the communication channels do not match. If $i \neq k \cap j = l$, from (14) we have

$$\begin{aligned} P_{PKGi}(I_n) \mid P_{PKGj}(I_m) \equiv{}& C_{PKGi}(x_1)_{\bullet} N_1(x_1)\, \overline{C_{PKGj}}\langle\sigma_1\rangle_{\bullet} C_{PKGi}(x_2)_{\bullet} N_2(x_2) \mid \\ & M_1(\sigma_1)\, \overline{C_{PKGk}}\langle y_1\rangle_{\bullet} C_{PKHl}(\theta_2)_{\bullet} M_2(\theta_2)\, \overline{C_{PKGk}}\langle y_2\rangle \end{aligned} \tag{15}$$

Since $i \neq k$, C_{PKGi} is different from C_{PKGk}. Thus, the protocol described in (15) cannot be executed successfully.

For $\{I_n = (i, j), I_m = (k, l) \mid I_n = I_m\}$, we have $P_{PKGi}(I_n) \mid P_{PKGj}(I_m) \equiv P_{PKGi}(i, j) \mid P_{PKGj}(i, j)$. In that case, the security protocol can be executed successfully. □

Proposition 2. For any group of process functions, if the channels of communication sides match mutually, the secrecy property holds.

Proof. According to (36)-(42) and Proposition 1, equation (12) can be transformed as:

$$\begin{aligned} Sys(I_1, I_2, \cdots I_n) \equiv & \prod_{i\in N}(vK^{-}_{PKGi})\prod_{j\in N}(vK^{-}_{PKGj})(P_{PKG}(I_1) \mid P_{PKG'}(I_1)) \\ & (P_{PKG}(I_2) \mid P_{PKG'}(I_2))\cdots(P_{PKG}(I_n) \mid P_{PKG'}(I_n)) \end{aligned} \tag{16}$$

Choose an arbitrary group of process functions from (16), based on (1)-(3) we have

$$\begin{aligned} & P_{PKG}(I_n) \mid P_{PKG'}(I_n) \equiv (vK^{-}_{PKGi})(vK^{-}_{PKGj})\,\overline{C_{PKGj}}\left\langle\{[(ID_{PKGi}, R_{i1,j})]\}_{ID_{PKGj},P}\right\rangle\cdots in\, F_n(x_{i41,j}, x_{i42,j}, x_{i43,j}) \mid \\ & C_{PKGj}(y_{i,j1})\cdots in\,\overline{C_{PKGi}}\left\langle\{[(P_j, [\{P_j\}]_{K^{-}_{PKGj},P}, [\{y_{i,j21}\}]_{K^{-}_{PKGj},P}, y_{i,j45})]\}_{y_{i,j21},P}\right\rangle \rightarrow\cdots\rightarrow \\ & (vK^{-}_{PKGi})(vK^{-}_{PKGj})\,F_n(P_j, [\{P_j\}]_{K^{-}_{PKGj},P}, [\{ID_{PKGi}\}]_{K^{-}_{PKGj},P}) \end{aligned} \tag{17}$$

where F_n is the PKG's private process without revealing any information about P_j, $[\{P_j\}]_{K^{-}_{PKGj},P}$ and $[\{ID_{PKGi}\}]_{K^{-}_{PKGj},P}$ in the event $I_n = (i, j)$. For (17), we have

$$\forall D, D' : F_n(D) \cong F_n(D') \Rightarrow P_{PKG}(i, j) \mid P_{PKG'}(i, j)\, with\, D \cong P_{PKG}(i, j) \mid P_{PKG'}(i, j)\, with\, D' \tag{18}$$

where D and D' represent different parameter collections.

According to (53), the confidentiality of the protocol process in (17) holds. □

Proposition 3. If the arbitrary group of process functions is secure, then for the whole protocol, which consists of groups of process functions, the secrecy property holds.

Proof. Since arbitrary group of process functions in (16) satisfies the indistinguishable relationship, we have

$$\begin{aligned} & \forall((D_1, D_1'), (D_2, D_2'), \cdots, (D_n, D_n')) : F_1(D_1) \cong F_1(D_1') \mid \cdots \mid F_n(D_n) \cong F_n(D_n') \Rightarrow \\ & (P_{PKG}(I_1) \mid P_{PKG'}(I_1)\, with\, D_1 \cong P_{PKG}(I_1) \mid P_{PKG'}(I_1)\, with\, D_1') \mid \cdots \mid (P_{PKG}(I_n) \mid P_{PKG'}(I_n) \\ & with\, D_n \cong P_{PKG}(I_n) \mid P_{PKG'}(I_n)\, with\, D_n') \end{aligned} \tag{19}$$

For arbitrary $((I_n, D_n), (I_n, D_n'))$, $P_{user}(I_n) \mid P_{SP}(I_n)\, with\, D_n \cong P_{user}(I_n) \mid P_{SP}(I_n)\, with\, D_n'$ holds. Based on (19) and the Proposition 1, for (12) we have

$$\begin{aligned} & \forall((I_1, D_1), (I_1, D_1')), ((I_2, D_2), (I_2, D_2')), \cdots, ((I_n, D_n), (I_n, D_n'))) : \\ & F_1(D_1) \cong F_1(D_1') \mid \cdots \mid F_n(D_n) \cong F_n(D_n') \Rightarrow \prod_{i\in N}(vK^{-}_{PKGi})\prod_{j\in N}(vK^{-}_{PKGj}) \\ & (P_{PKG}(I_1) \mid P_{PKG'}(I_1)\, with\, D_1)\cdots(P_{PKG}(I_n) \mid P_{PKG'}(I_n)\, with\, D_n) \cong \prod_{i\in N}(vK^{-}_{PKGi})\prod_{j\in N} \\ & (vK^{-}_{PKGj})(P_{PKG}(I_1) \mid P_{PKG'}(I_1)\, with\, D_1')\cdots(P_{PKG}(I_n) \mid P_{PKG'}(I_n)\, with\, D_n') \end{aligned} \tag{20}$$

From (20), we know that the process function $Sys(I_1, I_2, \cdots I_n)$ defined in (3) meets the secrecy definition given in appendix. Thus, the secrecy property of the protocol holds. □

5.1.2 Secrecy Analysis of User Access Authentication Protocol in RSD

We prove the secrecy property of the user access authentication protocol as follows:

Step 1. According to the proving procedures in Proposition 1, in the same way, we have that any group of protocol process functions composed of $P_{user-A}(I_n)$, $P_{SP-B}(I_m, D_m)$ and $P_{PKG-B}(I_k)$ can be executed successfully only if $I_n = I_m = I_k$ and the channels of communication sides match mutually.

Step 2. According to the structural equivalence axioms given in (36)-(42) and the conclusion in Step 1, we apply an equivalent transformation in (7) and obtain:

$$\begin{aligned} & Sys(I_1, I_2, \cdots I_n) \equiv (vK^{-}_{PKG-B})\prod_{i\in N}(vK^{-}_{useri-A})\prod_{j\in N}(vK^{-}_{SPj-B}) \\ & (P_{user-A}(I_1) \mid P_{SP-B}(I_1, D_1) \mid P_{PKG-B}(I_1))\cdots(P_{user-A}(I_n) \mid P_{SP-B}(I_n, D_n) \mid P_{PKG-B}(I_n)) \end{aligned} \tag{21}$$

In the process of protocol execution, since PKG-B (as an initiator) needs to obtain $(P_A,[\{P_A\}]_{K^-_{PKG-A},P},[\{ID_{PKG-B}\}]_{K^-_{PKG-A},P})$ from PKG-A (as a responder) and to provide P_A and $[\{ID_{PKG-B}\}]_{K^-_{PKG-A},P}$ for SP-B, equation (21) must be combined with (17) to form the complete security protocol expression (22).

$$\begin{aligned} &Sys(I_1,I_2,\cdots I_n)\equiv(vK^-_{PKG-A})(vK^-_{PKG-B})(P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B))\prod_{i\in N}(vK^-_{useri})\prod_{j\in N}(vK^-_{SPj})\\ &(P_{user-A}(I_1)\mid P_{SP-B}(I_1,D_1)\mid P_{PKG-B}(I_1))\cdots(P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)) \end{aligned} \tag{22}$$

Equation (22) shows that PKGs in domain A and B will execute mutual authentication and data transfer protocol in the case that users in domain A access SPs in domain B. The process $(P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B))$ can be combined with $(P_{user}(I_n)\mid P_{SP}(I_n,D_n)\mid P_{PKG}(I_n))$ to form a complete user access authentication protocol in RSD. According to the protocol process defined in (4)-(6) and structural equivalence axioms (36)-(42), the formal expression of the security protocol can be expressed as:

$$\begin{aligned} &P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-A}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B)\equiv(vIK_{i,j})(vCK_{i,j})\\ &(\overline{C_{SPj-B}}\langle(ID_{useri-A},RAI)\rangle_{\bullet}C_{useri-A}(x_{i1,j})_{\bullet}P_1(x_{i1,j})\mid C_{SPj-B}(y_{i,j1})_{\bullet}Q_1(y_{i,j1})\mid C_{PKG-B}(z_{j,k1})_{\bullet}R_1(z_{j,k1})_{\bullet}\\ &(\overline{C_{PKG-A}}\langle\{[(ID_{PKG-B},RAND_{B1,A})]\}_{ID_{PKG-A},P}\rangle_{\bullet}C_{PKG-B}(u_{B1,A})_{\bullet}G_1(u_{B1,A})\mid C_{PKG-A}(v_{B,A1})_{\bullet}H_1(v_{B,A1}))_{\bullet}\overline{C_{SPj-B}}\langle\sigma\rangle \end{aligned} \tag{23}$$

We apply an equivalent transformation in (23) and obtain:

$$\begin{aligned} &P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B)\equiv(vIK_{i,j})(vCK_{i,j})(C_{useri-A}(x_{i1,j})_{\bullet}\\ &P_1(x_{i1,j})\mid Q_1((ID_{useri},RAI)/y_{i,j1})\mid C_{PKG-B}(z_{j,k1})_{\bullet}R_1(z_{j,k1})_{\bullet}(\overline{C_{PKG-A}}\langle\{[(ID_{PKG-B},RAND_{B1,A})]\}_{ID_{PKG-A},P}\rangle_{\bullet}\\ &C_{PKG-B}(u_{B1,A})_{\bullet}G_1(u_{B1,A})\mid C_{PKG-A}(v_{B,A1})_{\bullet}H_1(v_{B,A1}))_{\bullet}\overline{C_{SPj-B}}\langle\sigma\rangle \end{aligned} \tag{24}$$

where $Q_1(y_{i,j1})\equiv let(y_{i,j11},y_{i,j12})=y_{i,j1}\ in[y_{i,j12}\ is\ not\ RAI_B]$

$\overline{C_{PKG-B}}\langle(\{[(y_{i,j1},RAND_{i,j1})]\}_{ID_{SPj},P_B},ID_{SPj}\rangle_{\bullet}C_{SPj-B}(y_{i,j2})_{\bullet}Q_2(y_{i,j2})$.

We substitute $Q_1(y_{i,j1})$ into (24) and obtain:

$$\begin{aligned} &P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B)\equiv(vIK_{i,j})(vCK_{i,j})(C_{useri-A}(x_{i1,j})_{\bullet}P_1(x_{i1,j})\\ &\mid C_{SPj-B}(y_{i,j2})_{\bullet}Q_2(y_{i,j2})\mid R_1((\{[(ID_{useri},RAI,RAND_{i,j1})]\}_{ID_{SPj},P_B},ID_{SPj})/z_{j,k1})_{\bullet}\\ &(\overline{C_{PKG-A}}\langle\{[(ID_{PKG-B},RAND_{B1,A})]\}_{ID_{PKG-A},P}\rangle_{\bullet}C_{PKG-B}(u_{B1,A})_{\bullet}G_1(u_{B1,A})\mid C_{PKG-A}(v_{B,A1})_{\bullet}H_1(v_{B,A1}))_{\bullet}\overline{C_{SPj-B}}\langle\sigma\rangle \end{aligned} \tag{25}$$

where $R_1(z_{j,k1})\equiv let(z_{j,k11},z_{j,k12})=z_{j,k1}\ in\ [z_{j,k12}\ is\ ID_{SPj}]\ case\ z_{j,k11}\ of\ \{[z_{j,k2}]\}_{K^-_{SPj},P_B}\ in$
$let(z_{j,k21},z_{j,k22},z_{j,k23})=z_{j,k2}\ in\ [z_{j,k22}\ is\ RAI_A]$.

From (23) and (A.2) - (A.25), we substitute $R_1(z_{j,k1})$ into (25) and obtain:

$$\begin{aligned} &P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B)\equiv(vIK_{i,j})(vCK_{i,j})\\ &(C_{useri-A}(x_{i1,j})_{\bullet}P_1(x_{i1,j})\mid Q_2(\{[(P_A,[\{P_B\}]_{K^-_{PKG-B},P_B},[\{ID_{PKG-B}\}]_{K^-_{PKG-A},P_A},RAND_{i,j1})]\}_{ID_{SPj},P_B}/y_{i,j2}) \end{aligned} \tag{26}$$

We use the corresponding process functions defined in (4) and (5) to substitute for $P_1(x_{i1,j})$ and $Q_2(y_{i,j2})$ in (26), and apply an equivalent transformation in (26) repeatedly until we get the final result:

$$P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B)\equiv(vIK_{i,j})(vCK_{i,j})F_n(D'_n) \tag{27}$$

where F_n is the SP's private process, which does not reveal any information about D_n in the event $I_n=(i,j)$. For (27) we have:

$$\begin{aligned} &\forall D_n,D'_n:F_n(D_n)\cong F_n(D'_n)\Rightarrow P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid\\ &P_{PKG-B}(A,B)\cong P_{user-A}(I_n)\mid P_{SP-B}(I_n,D'_n)\mid P_{PKG-B}(I_n)\mid P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B) \end{aligned} \tag{28}$$

Step 3. Since the arbitrary group of process functions in (29) satisfies the indistinguishable relationship, we have:

$$\begin{aligned}&\forall((I_1,D_1),(I_1,D_1')),((I_2,D_2),(I_2,D_2')),\cdots,((I_n,D_n),(I_n,D_n'))):F_1(D_1)\cong F_1(D_1')\mid\\&F_2(D_2)\cong F_2(D_2')\mid\cdots\mid F_n(D_n)\cong F_n(D_n')\Rightarrow(v\bar{K}_{PKG-A})(v\bar{K}_{PKG-B})\prod_{i\in N}(v\bar{K}_{useri-A})\prod_{j\in N}(v\bar{K}_{SPj-B})\\&(P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B))(P_{user-A}(I_1)\mid P_{SP-B}(I_1,D_1)\mid P_{PKG-B}(I_1))\cdots(P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n)\\&\mid P_{PKG-B}(I_n))\cong(v\bar{K}_{PKG-A})(v\bar{K}_{PKG-B})\prod_{i\in N}(v\bar{K}_{useri-A})\prod_{j\in N}(v\bar{K}_{SPj-B})(P_{PKG-A}(A,B)\mid P_{PKG-B}(A,B))\\&(P_{user-A}(I_1)\mid P_{SP-B}(I_1,D_1')\mid P_{PKG-B}(I_1))\cdots(P_{user-A}(I_n)\mid P_{SP-B}(I_n,D_n')\mid P_{PKG-B}(I_n))\end{aligned}\tag{29}$$

From (28), we know that the process function $Sys(I_1,I_2,\cdots I_n)$ defined in (7) satisfies the secrecy definition in appendix. Thus, the secrecy property of the access authentication and key agreement protocol holds. □

5.2 Authenticity Analysis

In the processes of our protocols, the knowledge that the attacker gets from communication channels can be expressed as $K=\{ID,RAI,C,D\}$, where ID represents the set of identity information, RAI the set of register area identifiers, C the set of channel identifiers and D the set of encrypted data.

From Section 5.1, we can get a conclusion that the attacker cannot attack the protocol even if he gets C and RAI. In our security architecture, zero-knowledge proof technique makes the attacker's masquerade ineffective, because the attacker cannot complete the response process without the legitimate private key. By state exploration approach [8], we can prove that the authenticity of the protocols relies on the freshness of random numbers. If the random number is fresh, the protocols satisfy $\forall M: Inst(M)\cong Inst_{spec}(M')$ and can resist man-in-the-middle attacks and masquerading attacks.

6. Experiment

In this section, we discuss the execution efficiency of the security protocols in Section 4. Firstly, we evaluate the execution efficiency of the algorithms in the BF-IBE scheme because they affect the performance of the protocols greatly due to their slow execution speed. Secondly, we test the AETs of the protocols under concurrent requests.

The hardware of our test environment includes a single Intel Core i5 quad-core processor, two DDR3 2G memory and 100Mbps bandwidth. The software environment includes Windows 7, Oracle JRE 1.7.1 and Web-logic 12C. The development environment includes Spring 3.1, Oracle JDK 1.7.1 and part of open source architecture from the third party.

In order to achieve the security protocols, we have implemented the algorithms of AES-128, HMAC-SHA1-160 and random-number-generation through Java Crypto API. Based on the open source code of BF-IBE algorithms in C offered by Stanford, we rewrote the code in Java to adapt to web service.

6.1 Performance Analysis of BF-IBE Algorithms

The BF-IBE scheme includes four algorithms: Setup algorithm generates system parameters and a master key; Extract algorithm uses the master key to generate the private key corresponding to an arbitrary public key; Encryption algorithm encrypts data using the public key and the system parameters; Decryption algorithm decrypts data using the private key and the system parameters.

The execution time of the setup algorithm is not discussed in this section, because this algorithm is only used for the system initialization and its performance does not affect the execution efficiency of the protocols. Moreover, compared with the encryption and the decryption algorithms, the signature and the verification algorithms can scarcely affect the execution time of our security protocols due to their high efficiency. So the test results of the signature and the verification algorithms are also not discussed.

6.1.1 Performance Analysis of the Extract Algorithm

In order to evaluate the impact of the extract algorithm on the time spent in the cloud entity registration and authorization, we select five sets of data to test the AETs of the extract algorithm, and each set contains 500 samples of public key with the same size. The test results are described in **Table 1**, under the condition that the public key size does not exceed 256 byte, the AETs are about 20ms. Due to the pre-authorization mechanism in our security scheme, the concurrent execution efficiency of the extract algorithm does not significantly affect the execution efficiency of the authentication protocols. Therefore, the concurrent test results are not discussed in this article.

Table 1. The average execution time of extract algorithm

Public Key Size (Byte)	Average Execution Time (ms)	Sample Amount	Private Key Size (bit)
16	19.9	500	512
32	20.1	500	512
64	20.0	500	512
128	20.4	500	512
256	20.3	500	512

6.1.2 Performance Analysis of Encryption and Decryption Algorithms

The encryption and the decryption algorithms are only used to process the protocol data of small amount (the maximum protocol packet size is smaller than 6KB). Thus, we select 10 sets of data with the size of each set ranging from 1KB to 10KB to test the AETs of the algorithms. Each set contains 500 data samples of the same size. As shown in **Fig. 6**, the AET of each algorithm is proportional to the data size (private key size is 512bit). When the data size is 6KB, the average encryption time is shorter than 0.2s and the decryption time is about 0.9s.

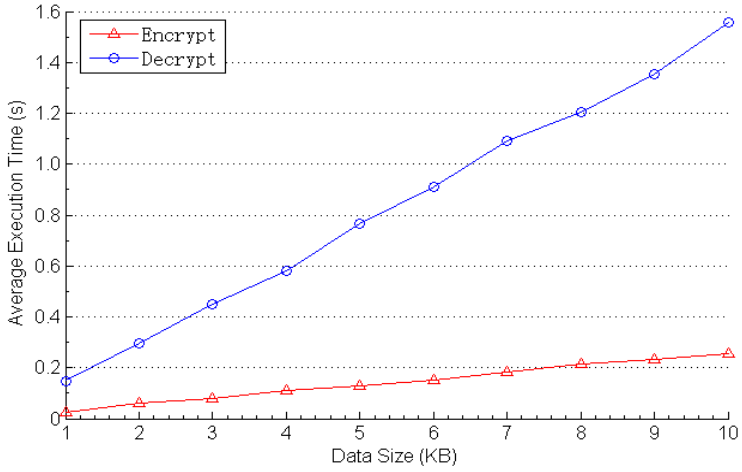

Fig. 6. Average execution times of encryption and decryption algorithms

According to the principle of the BF-IBE scheme, the public key can be any string and the

size of the public key does not affect the speed of the encryption and the decryption algorithms, while the size of the corresponding private key is fixed and determined by the key size parameter specified in the extract algorithm. Therefore, we need to evaluate the relationship between the execution time of the encryption or the decryption algorithm and the size of the private key. We select four private keys with the size of 128 bits, 256 bits, 512 bits and 1024 bits respectively, and test five sets of test data ranging from 2KB to 10KB to determine the AETs of both algorithms.

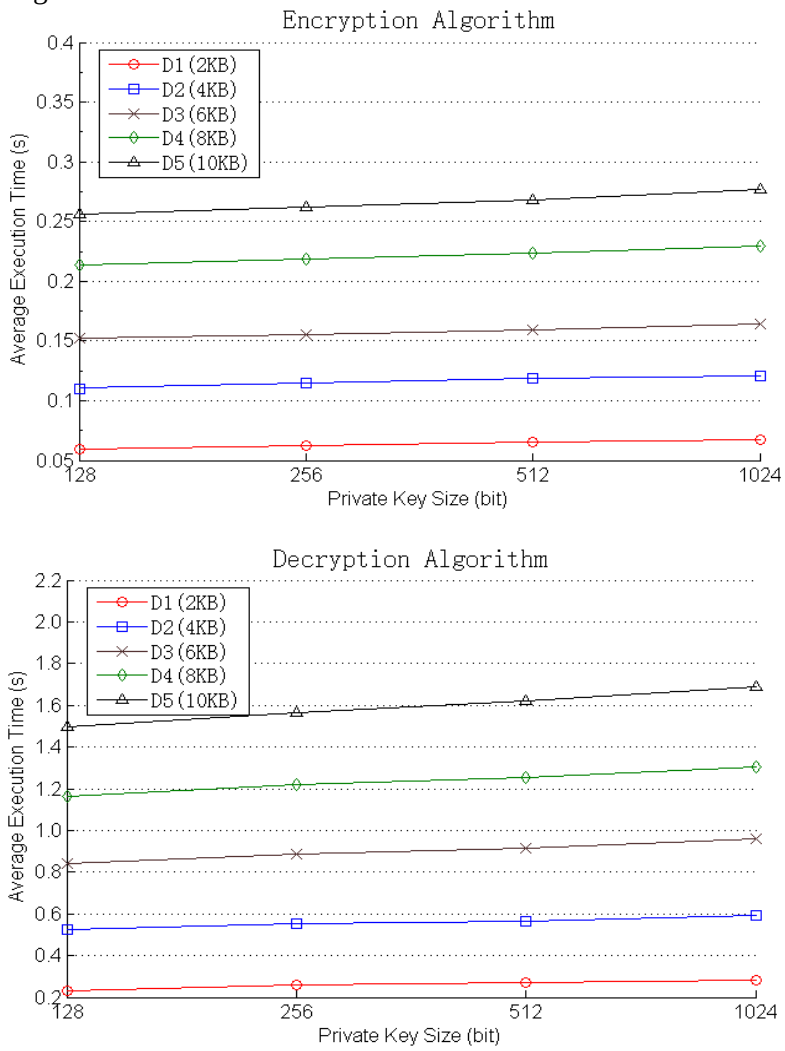

Fig. 7. Average execution times of encryption and decryption algorithms with different private key size

As shown in **Fig. 7**, the test results show that the AETs of the encryption and the decryption algorithms are proportional to the size of the private key and the growth rates of the AETs are proportional to the size of the processed data. In the case of small amount of data, the growth of the execution time is no more than 0.2 seconds with the increase of private key size and the processed data. Therefore, the stability of the encryption and the decryption algorithms in the BF-IBE scheme achieves application requirements.

6.2 Protocol Performance Analysis

In order to evaluate the performance of our protocols, we test the AETs of four protocols under

concurrent requests (private key size is 512bit). The number of concurrent requests range from 10 to 100. In **Fig. 8**, P1 depicts the AET of mutual authentication protocol between PKGs in GSD, P2 the AET of user access authentication protocol in RSD, P3 the AET of mutual authentication protocol between SPs in different SDs and P4 the AET of data sharing protocol in different SDs.

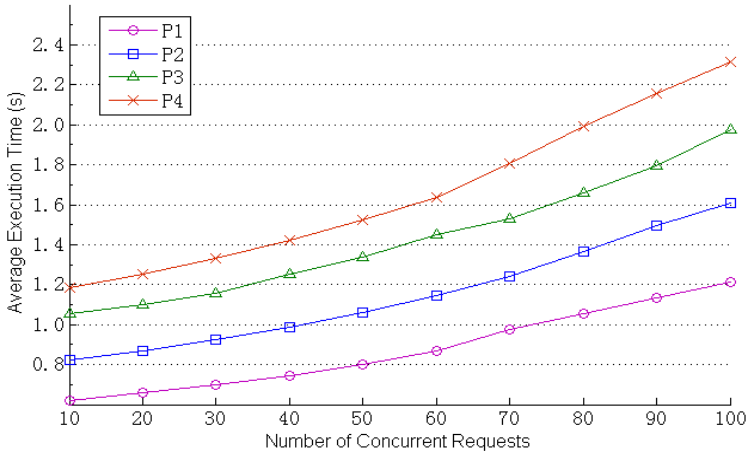

Fig. 8. Average execution times of four security protocols

As shown in **Fig. 8**, with the increase of the concurrent requirements, the execution time grows. When the concurrent requests are the same, the execution time of the protocols is proportional to the complexity and the packet size of the protocols. The longest execution time of the data sharing protocol is shorter than 2.4s, in the case of 100 concurrent requests. The AET of each security protocol is proportional to the number of times the encryption and the decryption algorithms are executed, and at the same time proportional to the size of the data to be encrypted or decrypted.

7. Conclusion

In order to solve cloud entity authentication issues, we divide the federated cloud system into multiple cloud units and each unit is a self-contained domain. On the basis of BF-IBE scheme, we propose a layered security architecture and four security protocols to implement mutual authentication, data sharing and secure data transfer in the federated cloud system. We use S-pi calculus formal verification to prove the security properties of the protocols and make the conclusion: these protocols not only protect the transferred sensitive data, but also resist man-in-the-middle attacks and masquerading attacks. The layered security architecture and the performance of the protocols are tested in a lab environment.

Compared to HIBE scheme, the layered architecture is more suitable to implement mutual authentication for cloud entities which belong to different SDs in the federated cloud system, because GPKG only provides trust transmission mechanism for different SDs but cannot decrypt the private information protected by the lower-level PKG. How to effectively incorporate the HIBE scheme into our security architecture in the LSD will be the future work.

Moreover, in the future research we will investigate the case of multiple Virtual Machines (VMs), in which the VM with the weakest security becomes the missing link for the whole system. Since traditional network security systems, such as firewalls and intrusion detection systems, may fail in the virtualized network, secure low-level VM to VM access will be a new

technical difficulty in constructing the federated cloud system. Our future work will focus on the security of VM access and re-imaged VMs.

Appendix

Based on S-pi calculus [7], [8] the semantic notations in this article are given in **Fig. 9**.

n	name	$\sigma(x).P$	input
$x, y, \cdots$	name(variable)	$P \mid Q$	composition
(σ, ρ, θ)	pair	$(vm)P$	restriction
$suc(\sigma)$	successor	$!P$	replication
$M(\sigma, \rho)$	MAC	$[\sigma \; is \; \rho]P$	match
$\{\sigma\}_\rho$	shared-key encryption	$[\sigma \; is \; not \; \rho]P$	not match
σ^+	public part	$let(x, y) = \sigma \; in \; P$	pair splitting
σ^-	private part	$case \; \sigma \; of \; 0 : P \; suc(x) : Q$	integer case
$\{[\sigma]\}_{\rho,\theta}$	public-key encryption	$case \; \sigma \; of \; \{x\}_\rho \; in \; P$	shared-key decryption
$[\{\sigma\}]_{\rho,\theta}$	private-key signature	$case \; \sigma \; of \; \{[x]\}_{\rho,\theta} \; in \; P$	decryption
$\overline{\sigma}\langle\rho\rangle.P$	output	$case \; \sigma \; of \; [\{x\}]_{\rho,\theta} \; in \; P$	signature check

Fig. 9. Syntaxes of S-pi calculus terms and processes

— $\{[\sigma]\}_{\rho,\theta}$ represents the encryption result of σ with public key ρ and public parameter θ.

— $[\{\sigma\}]_{\rho,\theta}$ represents the signature result of σ with private key ρ and public parameter θ.

— $M(\sigma, \rho)$ represents the message authentication code of σ with symmetric key ρ.

— $case \; \sigma \; of \; \{[x]\}_{\rho,\theta} \; in \; P$ represents a private key decryption process. If σ is the result of encrypting message μ with a public key whose corresponding private key is ρ and the public parameter is θ, the process substitutes x with μ and behaves as described in P.

— $case \; \sigma \; of \; [\{x\}]_{\rho,\theta} \; in \; P$ represents a signature check process. If σ is the signature of message μ signed with a private key whose corresponding public key is ρ and the public parameter is θ, the process substitutes x with μ and behaves as described in P.

Reduction relation axioms:

$$!P > P \mid !P \quad (30) \qquad [\sigma \; is \; \rho]P > P \quad (31) \qquad let(x, y) = (\sigma, \rho) \; in \; P > P[\sigma/x, \rho/y] \quad (32)$$

$$case \; \{\sigma\}_\rho \; of \; \{x\}_\rho \; in \; P > P[\sigma/x] \quad (33) \qquad case \; \{[\sigma]\}_{\rho^+,\theta} \; of \; \{[x]\}_{\rho^-,\theta} \; in \; P > P[\sigma/x] \quad (34)$$

$$case \; [\{\sigma\}]_{\rho^-,\theta} \; of \; [\{x\}]_{\rho^+,\theta} \; in \; P > P[\sigma/x] \quad (35)$$

Structural equivalence axioms:

$$P \equiv P \quad (36) \qquad P \mid (Q \mid R) \equiv (P \mid Q) \mid R \quad (37) \qquad P \mid 0 \equiv P \quad (38) \qquad (vm)(vn)P \equiv (vn)(vm)P \quad (39)$$

$$(vn)0 \equiv 0 \quad (40) \qquad P \mid Q \equiv Q \mid P \quad (41) \qquad (vn)(P \mid Q) \equiv P \mid (vn)Q \quad if \; n \notin fn(P) \quad (42)$$

Reaction relation axiom and rules:

$$\overline{\sigma}\langle\rho\rangle.P \mid \sigma(x).Q \to P \mid Q[\rho/x] \quad (43) \qquad \frac{P > Q}{P \equiv Q} \quad (44) \qquad \frac{}{P \equiv P} \quad (45) \qquad \frac{P \equiv Q}{Q \equiv P} \quad (46)$$

$$\frac{P \equiv Q \quad Q \equiv R}{P \equiv R} \quad (47) \qquad \frac{P \equiv P'}{P \mid Q \equiv P' \mid Q} \quad (48) \qquad \frac{P \equiv P'}{(vn)P \equiv (vn)P'} \quad (49) \qquad \frac{P \equiv P' \quad P' \to Q' \quad Q' \equiv Q}{P \equiv Q} \quad (50)$$

$$\frac{P \to P'}{P \mid Q \to P' \mid Q} \quad (51) \qquad \frac{P \to P'}{(vn)P \to (vn)P'} \quad (52)$$

Suppose that $Inst(M)$ expresses a security protocol and $Inst_{spec}(M)$ expresses a correct specification of the protocol. The secrecy property of the protocol can be defined as:

$$\forall M, M' : F(M) \cong F(M') \Rightarrow Inst(M) \cong Inst_{spec}(M') \quad (53)$$

The authenticity property of the protocol can be defined as:

$$\forall M : Inst(M) \cong Inst_{spec}(M') \quad (54)$$

References

[1] *The NIST Definition of Cloud Computing*, NIST Special Publication 800-145, September, 2011. Article (CrossRef Link).

[2] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Generation Computer Systems*, vol. 28, no. 3, pp. 583-592, March, 2012. Article (CrossRef Link).

[3] D. G. Guo, M. Zhang, Y. Zhang and Z. Xu, "Study on cloud computing security," *Journal of Software*, vol.22, no.1, pp.71−83, January, 2011. Article (CrossRef Link).

[4] S. Grzonkowski and P. M. Corcoran, "Sharing cloud services: user authentication for social enhancement of home networking," *IEEE Trans. Consumer Electron.*, vol. 57, no. 3, pp. 1424-1432, August, 2011. Article (CrossRef Link).

[5] S. Grzonkowski and P. M. Corcoran, "Security analysis of authentication protocols for next-generation mobile and CE cloud services," in *Proc. of 1st IEEE International Conf. Consumer Electron. Berlin*, pp. 83-87, 2011. Article (CrossRef Link).

[6] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586-615, 2003. Article (CrossRef Link).

[7] M. Abadi and A. D. Gordon "A calculus for cryptographic protocols: the Spi calculus," *Information and Computation*, vol. 148, no. 1, pp. 1-70, 1999. Article (CrossRef Link).

[8] L. Durante, R. Sisto, and A. Valenzano, "Automatic testing equivalence verification of Spi calculus specifications," *ACM Trans. Software Engineering and Methodology*, vol. 12, no. 2, pp. 222-284, April, 2003. Article (CrossRef Link).

[9] A. Tiu and J. Dawson, "Automating open bisimulation checking for the Spi calculus," in *Proc. of 23rd IEEE Computer Security Foundations Symposium*, pp. 307-321, 2010. Article (CrossRef Link).

[10] C. Gentry and A. Silverberg, "Hierarchical ID-based cryptography," in *Proc. of ASIACRYPT'02*, vol. 2501, pp. 548 – 566, 2002. Article (CrossRef Link).

[11] D. Boneh and X. Boyen, "Efficient selective-ID secure identity based encryption without random oracles," *Lecture Notes in Computer Science* vol. 3027, pp. 223-238, 2004. Article (CrossRef Link).

[12] D. Boneh, X. Boyen, and E. J. Goh, "Hierarchical identity based encryption with constant size ciphertext," *Lecture Notes in Computer Science*, vol. 3494, pp. 440-456, 2005. Article (CrossRef Link).

[13] C. Gentry and S. Halevi, "Hierarchical identity based encryption with polynomially many levels," *Lecture Notes in Computer Science*, vol. 5444, pp. 437-456, 2009. Article (CrossRef Link).

[14] D. Boneh and X. Boyen, "Efficient selective identity-based encryption without random oracles," *Journal of Cryptology*, vol. 24, no. 4, pp. 659-693, October, 2011. Article (CrossRef Link).

[15] H. W. Lim and K. G. Paterson, "Identity-based cryptography for grid security," *International Journal of Information Security*, vol. 10, no. 1, pp. 15-32, 2011. Article (CrossRef Link).

[16] H. Cheng, C.Rong, Z. Tan, and Q. Zeng, "Identity based encryption and biometric authentication scheme for secure data access in cloud computing," *Chinese Journal of Electronics*, vol. 21, no.

2, April, 2012. Article (CrossRef Link).
[17] H. W. Li, Y. S. Dai, T. Ling, and H. M. Yang, "Identity-based authentication for cloud computing," *Lecture Notes in Computer Science*, vol. 5931, pp. 157-166, 2009. Article (CrossRef Link).
[18] L. S. Kang and X. J. Zhang, "Identity-based authentication in cloud storage sharing," in Proc. *of 2nd International Conf. on Multimedia Information Networking and Security*, pp. 851-855, 2010. Article (CrossRef Link).
[19] L. Yan, C. M. Rong, and G.S. Zhao, "Strengthen cloud computing security with federal identity management using hierarchical identity-based cryptography," *Lecture Notes in Computer Science,* vol. 5931, pp. 167-177, 2009. Article (CrossRef Link).
[20] J. Y. Huang, I. E. Liao, and C. K. Chiang, "Efficient identity-based key management for configurable hierarchical cloud computing environment," in *Proc. of International Conf. on Parallel and Distributed Systems*, pp. 883-887, 2011. Article (CrossRef Link).
[21] C. Schridde, M. Smith, and B. Freisleben, "An identity-based key agreement protocol for the network layer," *Lecture Notes in Computer Science*, vol. 5229, pp. 409-422, 2008. Article (CrossRef Link).
[22] C. Schridde, T. Dörnemann, E. Juhnke, B. Freisleben, and M. Smith, "An identity-based security infrastructure for cloud environments," in *Proc. of IEEE International Conf. on Wireless Communications, Networking and Information Security*, pp. 644-649, 2010. Article (CrossRef Link).
[23] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," *Lecture Notes in Computer Science*, vol. 6054, pp. 136-149, 2010. Article (CrossRef Link).
[24] X. Yang, B. Nasser, M. Surridge, and S. Middleton, "A Business-oriented cloud federation model for real-time applications," *Future Generation Computer Systems*, vol. 28, pp. 1158–1167, October, 2012. Article (CrossRef Link).
[25] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin et al., "The RESERVOIR model and architecture for open federated cloud computing," *IBM Journal of Research & Development*, vol.53 (4), pp.535–545, 2009. Article (CrossRef Link).
[26] D. Villegas, N. Bobroff, I. Roderob, J. Delgado, Y. Liu et al., "Cloud federation in a layered service model," *Journal of Computer and System Sciences*, vol. 78, pp. 1330–1344, September, 2012. Article (CrossRef Link).
[27] C. Neuman, T. Yu, S. Hartman and K. Raeburn, "The Kerberos network authentication service (V5)," *http://www.ietf.org/rfc/rfc4120*, July, 2005. Article (CrossRef Link).
[28] B. Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C*, 2nd Edition, John Wiley & Sons, 1996. Article (CrossRef Link).
[29] D. Recordon and B. Fitzpatrick, "OpenID authentication 2.0," *http://openid.net/specs/openid-authentication-2_0.html*, December, 2007. Article (CrossRef Link).
[30] S. Sun, K. Hawkey, and K. Beznosov, "Systematically breaking and fixing OpenID security: Formal analysis, semi-automated empirical evaluation, and practical countermeasures," *Computers & Security*, vol.31, pp. 465-483, June, 2012. Article (CrossRef Link).

Chenlei Cao was born in 1982. He received his B.S. degree in Electronic Information Engineering in 2005 and M.S. degree in Information Security from Beijing University of Post and Telecommunications (BUPT) in 2009. He is currently a Ph.D. student in Information Security at BUPT. His interests include system security, cryptography and cloud computing.

Ru Zhang was born in 1976 and received Ph.D. degree in Computer Application Technology in 2003. She is a Prof. at Computer College, BUPT. She researches on Information Security in the state key laboratory of networking and switching technology, BUPT. Her interests include digital watermark, cryptography and multimedia authentication. She was awarded a national second prize and two provincial prizes. She is the author of more than 20 publications and 3 books. She holds four state patents.

Mengyi Zhang was born in 1988 and received B.S. degree in Information Security from BUPT in 2010. She is presently a master student in Information Security, BUPT. Her research interests include system security, cryptography and cloud computing.

Yixian Yang was born in 1961 and received Ph.D. degree in Electronics and Communication Systems in 1988 from BUPT. He is a Prof. at Computer College, BUPT. He researches on Information Security in the state key laboratory of networking and switching technology, BUPT. His interests include digital watermark, cryptography and multimedia authentication.