

SECURE SHARING OF SENSITIVE DATA ON ABIG DATA PLATFORM USING HOMOMORPHIC SESSION BASED PROXY RE-ENCRYPTION

Ms.K.P.AMMU, Mrs.B.RASINA BEGUM

¹Post Graduate Student, ²Assistant professor, Department of Computer Science and Engineering
Mohamed Sathak Engineering College, Kilakarai, Tamilnadu, India.
ammukp01@gmail.com

Abstract -Big data presents a tremendous opportunity for enterprises across industries. By tapping into new volumes and varieties of data, scientists, executives, product managers, marketers, and a range of others can start making more informed plans and decisions, discover new opportunities for optimization, and deliver breakthrough innovations. Here the challenge is data sharing on big data platform with security is more problematic. To overcome the challenges we propose a framework for the secure data sharing which includes secure data delivery, storage, usage and destruction on a big data platform. Here we propose the Homomorphic time based proxy re-encryption algorithm and a user process protection method based on a virtual machine monitor that provides data protection on user process. The data owner encrypts sensitive data using local security plug-in concept and data self-destruction mechanism helps to alleviate user concern regarding sensitive personal information leakage. Our framework allows the sensitive data sharing securely and the data owners retain complete control of their own data in a sound environment for modern internet information security. The entire system studies the problem of ensuring the secure sensitive data sharing and greatly reduces the overhead of the interaction among tangled parties.

Keywords: Secure Data Sharing, Data Leakage, Homomorphic Session Based Proxy Re-encryption, Security Plug-in, etc.

I. INTRODUCTION

With the prompt development of information digitization, massive amounts of structured, semi-structured, and unstructured data are generated rapidly. By collecting, sorting, analyzing, and mining these data, an enterprise can attain vast amounts of individual users' sensitive data. These data not only meet the demands of the enterprise itself, but also provide services to other companies if the data are stored on a big data platform. Traditional cloud storage merely stores plain text or encrypted data passively. However, a big data platform allows the exchange of data (including sensitive data). It provides mass data storage and computational services. Computation services refer primarily to operations on data used by participants, which can invigorate "dead" data.

We consider user's preferences as sensitive data. When Alice submits a query. The Search Engine Service Provider (SESP) first looks for Alice's preference on the big data platform. If the big data platform has collected and shared the user's personal preference information. Then the search engine returns personalized results(sportswear + badminton) to Alice. When Alice sees her favorite badminton sportswear, she experiences a pleasant purchase. However, while data sharing increases enterprise assets, Internet

insecurity and the potential of sensitive data leakage also create security issues for sensitive data sharing.

A big data platform is a complete system with multistakeholder involvement, and thus cannot tolerate any security breach resulting in sensitive data loss. Sensitive data is defined as information that is protected against unwarranted disclosure. Access to sensitive information should be protected. Protection of sensitive information may be required for legal or ethical reasons, for issues pertaining to personal privacy, or for proprietary considerations. Proxy re-encryption schemes are cryptosystems which allow third parties (proxies) to alter a cipher text which has been encrypted for one party, so that it may be decrypted by another. A proxy re-encryption is generally used when one party, that wants to reveal the contents of messages sent to him and encrypted with his public key to a third party, Third party, without revealing his private key to . First party does not want the proxy to be able to read the contents of his messages. First party could designate a proxy to re-encrypt one of his messages that is to be sent to Third party. This generates a new key that Third party can use to decrypt the message. Now if Alice sends Third party a message that was encrypted under First party's key, the proxy will alter the message, allowing Third party to decrypt it. This method allows for a

number of applications such as e-mail forwarding, law-enforcement monitoring, and content distribution. A weaker re-encryption scheme is one in which the proxy possesses both parties' keys simultaneously. One key decrypts a plaintext, while the other encrypts it. Since the goal of many proxy re-encryption schemes is to avoid revealing either of the keys or the underlying plaintext to the proxy, this method is not ideal. To overcome the challenges we propose a framework for the secure data sharing which includes secure data delivery, storage, usage and destruction on a big data platform. Here we propose the Homomorphic time based proxy re-encryption algorithm and a user process protection method based on a virtual machine monitor that provides data protection on user process.

In this paper, we propose the data sharing sensitively and securely for the data owners who share their sensitive data to their neighbor's or friends. Addressed some secure approaches for overcoming the dilemma in security on big data platform.

Request Alice preferences (Alice and bob's ID)

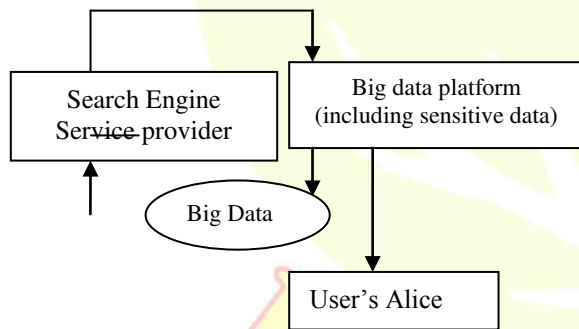


Fig.1 Application of sensitive data (user's preferences)

II. RELATED WORK

Data encryption and access control of big data Storage:

The Attribute-Based Encryption (ABE) algorithm includes Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CPABE) ABE decryption rules are contained in the encryption algorithm, avoiding the costs of frequent key distribution in ciphertext access control. When the access control strategy changes dynamically, a data owner is required to re-encrypt the data. A semi-trusted agent with a proxy key can re-encrypt Ciphertext; however, the agent cannot obtain the corresponding plaintext or Compute the decryption key of either party in the authorization process. The FHE mechanism permits a specific algebraic operation based on ciphertext that yields a still encrypted result. More specifically, retrieval and comparison of the encrypted data produce correct results, but the data are not decrypted throughout the entire process. The FHE scheme

requires very substantial computation, and it is not always easy to implement with existing technology. In regard to Ciphertext retrieval with a view toward data privacy

Protection in the cloud, ciphertext retrieval solutions in the cloud are proposed in Refs. [6–8]. Regarding access control, a new cryptographic access control scheme, Attribute-Based Access Control for Cloud Storage (ABACCS), is proposed in Ref. [9]. Each user's private key is labeled with a set of attributes, and data is encrypted with an attribute condition restricting the user to be able to decrypt the data only if their attributes satisfy the data's condition. Distributed systems with Information Flow Control (DIFC)[10] use a tag to track data based on a set of simple data tracking rules. DIFC allow untrusted software to use private data, but use trusted code to control whether the private data can be revealed. In Ref. [11], the authors consider the complexity of fine-grained access control for a large number of users in the cloud and propose a secure and efficient revocation scheme based on a modified CP-ABE algorithm. This algorithm is used to establish fine-grained access control in which users are revoked.

Trusted computing and process protection:Trusted Computing Group (TCG) introduced the Trusted Platform Module (TPM) in its existing architecture, to ensure that a general trusted computing platform using TPM security features is credible. The main research idea includes first building a trusted terminal platform based on a security chip, and then establishing trust between platforms through remote attestation. Then, trust is extended to the network. TC Gissued a Trusted Network Connection (TNC) architecture specification version in 2005, characterized by having terminal integrity as a decision of network access control. Feng et al.[16] proposed a trustworthiness-based trust model and provided a method of building a trust chain dynamically with information flow. Zhang et al.[17] proposed a transparent, backward-compatible approach that protects the privacy and integrity of customers virtual machines on commodity virtualized infrastructures. Dissolver is a prototype system based on a VMM and Confidentiality and High-Assurance Equipped Operating System. It ensures that the user's text data exist only in a private operating space and that the user's key exists only in the memory space of the VMM. Data in the memory and the user's key are destroyed at a user-specified time.

Data destruction:Wang et al.[22] proposed a security destruction scheme for electronic data. A new scheme, Self Vanish, is proposed in Ref. [23]. This scheme prevents hopping attacks by extending the lengths of key shares and significantly increasing the cost of mounting an attack. To solve the problem of

how to prevent sensitive information from leaking, when an emergency occurs, Dong et al.[24] proposed a real-time sensitive safe data destruction system. The open source cloud computing storage system, Hadoop Distributed File System (HDFS), cannot destroy data completely, which may lead to data leak. To repair this flaw, Qin et al.[25] designed a multi-grade safe data destruction mechanism for HDFS. The authors proposed privacy management across the entire data lifecycle and used a mandatory data destruction protocol to control user data. As far as we know, few studies focus on the sharing of sensitive data on a big data platform. Here we provided a common framework for cataloguing and sharing both public and private data, but they do not discuss data computation on a big data platform. In this paper, we discuss the problem of data storage, computing, use, and destruction.

III. SYSTEMATIC FRAMEWORK FOR SECURE SENSITIVE DATA SHARING ON A BIG DATA PLATFORM

Issuing and renting sensitive data on a semi trusted big data platform requires a data security mechanism. Building secure channels for a full sensitive data life cycle requires consideration of four aspects of safety problems: reliable submission, safe storage, riskless use, and secure destruction.

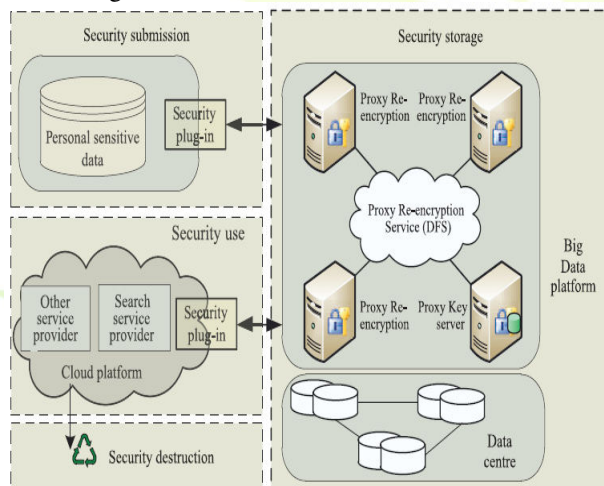


Fig.2 Systematic framework for secure sensitive data sharing on a big data platform.

A systematic framework for secure sensitive data sharing on a big data platform is shown in Fig. 2. A common and popular method of ensuring data submission security on a semi-trusted big data platform is to encrypt data before submitting data to the platform. Some operations (such as encryption, decryption, and authorization) are provided using a security plug-in. A cloud platform service provider (such as an SESP) using data on a big data platform

ensures data security by downloading and using the security plug-in. To ensure secure storage, we designed Homogeneous Proxy Re-Encryption (H-PRE), which supports homogeneous transformation from Identity-Based Encryption (IBE) to Public-Key Encryption (PKE). H-PRE is compatible with traditional cryptography. The intent is to transform the cipher data that the owner uploads into ciphertext that the data user can decrypt using his or her own private key. We assume that the cloud cannot be trusted, and that the decrypted clear text will leak users' private information. Therefore, we need to adopt process protection technology based on a VMM, through a trusted VMM layer, bypassing the guest operating system and providing data protection directly to the user process. The key management module of the VMM is used for storing public keys of the new register program group. When a program is running, the symmetric key at the bottom of the main program will be decrypted dynamically by the key management module. All applications of the public and symmetric keys are stored in the memory of the VMM. The archive, replication, and backup mechanism of cloud storage create data redundancy, requiring the use of a suitable data destruction scheme to delete the user's private personal data. To achieve high security, we designed a lease-based mechanism to destroy private data and keys thoroughly in a controlled manner. Clear text and keys exist nowhere in the cloud, after the lease expires.

The basic flow of the framework is as follows. First, Enterprises that have individual users' sensitive information pre-set those service providers that need to share this sensitive information and then submit and store the corresponding encrypted data on a big data platform using the local security plugin. Second, we need to perform the required operation with the submitted data using PRE on the big data platform. Then, cloud platform service providers who want to share the sensitive information download and decrypt the corresponding data in the private process space using the secure plug-in with sensitive privacy data running in that space. Last, we use a secure mechanism to destroy used data still stored temporarily in the cloud. In short, the framework protects the security of the full sensitive data life cycle effectively. Meanwhile, data owners have complete control over their own data. Next, we discuss the most critical PRE algorithm based on heterogeneous cipher-text transformation and user process protection methods using the VMM.

IV. SECURE SUBMISSION AND STORAGE OF SENSITIVE DATA BASED ON PRE H-PRE

H-PRE involves three types of algorithm, traditional identity-based encryption (including SetupIBE, KeyGenIBE, EncIBE, and DecIBE), re-encryption (including KeyGenRE, ReEnc, and ReDec functions), and the last one is the traditional public key cryptosystems (including KeyGenPKE, EncPKE, and DecPKE). The basic H-PRE process is simple. The data owner encrypts sensitive data using a local security plug-in and then uploads the encrypted data to a big data platform. The data are transformed into the ciphertext that can be decrypted by a specified user after PRE services. If an SESP is the specified user, then the SESP can decrypt the data using its own private key to obtain the corresponding clear text. We complete the following steps to implement the H-PRE algorithm.

- (1) Setup_{IBE}(k): Input security parameters k, generate randomly a primary security parameter mk, calculate the system parameter set params using a bilinear map and hash function.
- (2) KeyGen_{IBE}(mk, params, id): When the user requests the private key from the key generation center, the key generation center obtains the legal identity (id) of the user and generates the public and private keys (pk_{id}, sk_{id}) for the user using params and mk.
- (3) KeyGen_{PKE}(params): When a user submits a request, the key management center not only generates the identity-based public and private keys, but also generates the public and private keys of the traditional public key system (pk_{id}, sk_{id}).
- (4) Enc_{IBE}(pk_{id}, sk_{id}, params, m): When the user encrypts data, the data owner encrypts the clear-text (m) into the ciphertext (c = c₁, c₂) using the user's own (pk_{id}, sk_{id}) and a random number.
- (5) KeyGen_{RE}(sk_{id}, sk_{0id}, pk_{idj}, params): When the data owner (user i) grants user j permissions, using sk_{id}, sk_{0id}, pk_{idj}, params user i computes the PRE key completing the transformation from user i to user j.
- (6) ReEnc(c_i, rk_{idj}, params): This process is executed transparently on the big data platform. The function re-encrypts the ciphertext that user i encrypted into ciphertext that user j can decrypt. It inputs c_i (c_i=c_{i1}, c_{i2}), the PRE key, and related system parameters, and then the big data platform computes and outputs the PRE ciphertext (c_j=c_{j1}, c_{j2}).
- (7) Dec_{PKE}(c_j, sk₀, id_j, params): This is a function for decrypting the PRE ciphertext. After receiving the PRE ciphertext (c_j=c_{j1}, c_{j2}) from the proxy server of the big data platform, user j determines the clear-text of the data using his or her own sk₀, id_j.

The submission, storage, and extraction operations of system sensitive data:

The data owner encrypts data locally, first using the Advanced Encryption Standard (AES) symmetric encryption algorithm to encrypt the submission data

and then using the PRE algorithm to encrypt the symmetric key of the data. These results are all stored within the distributed data. In the meantime, if the data owner shares the sensitive data with other users, the data owner must authorize the sensitive data locally and generate the PRE key, which is stored in the authorization key server. On the big data platform, the PRE server re-encrypts and transforms the original cipher using the PRE key. Then, PRE ciphertext, which can be encrypted by the (authorized) data users, is generated.

If the data user wants to use the data on the big data platform, the data user will send data requests to the platform and then query whether there is corresponding data in the shared space. If such data exist, the data user accesses and downloads it. The operation on the big data platform is independent and transparent to users. The PRE system includes data submission, storage (sharing), and data extraction operations. Data submission and storage (sharing) operations After receiving the data uploading request, the Web browser invokes the security plug-in and provides data uploading services for the data owner, in accordance with the following detailed steps. The browser (1) reads the data files uploaded by the data owner generates randomly an AES transparent encryption key (Symmetric Encryption Key, SEK), and then use the AES algorithm to encrypt the data files; (2) uses the PRE algorithm to encrypt the SEK and store the data ciphertext and SEK ciphertext in the data centers; (3) identifies from the data owner the users designated to share the data; (4) uses the security plug-in to read the private key of the data owner and obtain the data user's public key from the big data platform; (5) uses the security plug-in to generate the corresponding PRE key using the EncIBE function and to upload the PRE key to the authorization key server of the big data platform; and (6) re-encrypts the data using the ReEnc function on the big data platform, thereby generating PRE ciphertext. Data extraction operations After receiving the data download request, the Web browser invokes the security plug-in and provides data download services for the data user, in accordance with the following detailed steps. The browser (1) queries whether there is authorization for the data user on the PRE server of the big data platform, and if an authorization is in effect, proceeds to Step (2); (2) uses the download plug-ins to send data download requests to the big data platform, which then finds PRE ciphertext data in the data center; (3) pushes the PRE ciphertext to the secure data plug-in on the big data platform; (4) invokes a data user's download plug-in to read the user's private key and prepares to decrypt data; (5) invokes a data user's download plug-in to decrypt

received SEK ciphertext using the Dec_{PKE} function and obtain the AES symmetric key; and (6) permits the data user to decrypt the data ciphertext using the AES symmetric key to obtain the required clear text. The data extraction operation is put into the private space of a user process by the secure plug-in, a prerequisite for secure use of sensitive data.

V. SECURE USES OF SENSITIVE DATA ON VMM

The private space of a user process based on a VMM

To ensure secure running of an application in the cloud, we use the private space of a user process based on a VMM. We assume that some enterprise (such as an SESP) rents Infrastructure as a Service (IaaS) to complete some business. The business process needs to extract sensitive personal data on the big data platform. We call the protected program that extracts sensitive data from the big data platform a sensitive process. A threat model of a sensitive process on a cloud platform is shown in Fig. 3. A sensitive process must prevent threats from a management VMM and an unreliable operating system layer below it. Rented bottom hardware uses the TPM mode, ensuring that the VMM is trusted. In this case, the key management mechanism of the renter (such as an SESP) must build this relationship based on trusting a VMM, ensuring safe operation under the unreliable operating system. The introduction of virtualization and trusted computing technology ensures that service provider applications and a secure plug-in run in the process private space. This mode can protect the privacy of sensitive data and avoid interference from external programs, even the operating system. A safe operation Fig. 3 Threat model of a sensitive process in a cloud platform. With PRE ciphertext calculated on a big data platform extracted onto a cloud platform, private memory space of processes on the cloud platform can guarantee data security in memory and on the Hard Disk Drive (HDD).

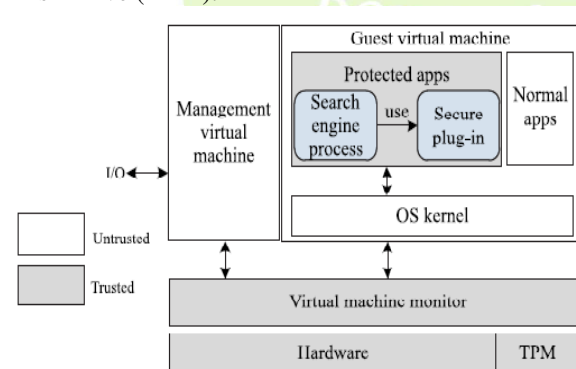


Fig.3 Threat model of a sensitive process in a cloud platform.

First, the VMM provides private memory space for specifying a VM process. The process runs in private memory space whose memory cannot be accessed by the operating system or other applications. The method of memory isolation ensures data privacy and security in the memory. Furthermore, the data used and stored on disk is ciphertext. The VMM decrypts or encrypts when reading or writing data, respectively. As a result, a combination of these two measures can be protected using the VMM, whether the user program runs in memory or is stored on disk.

(1) Secure use of system sensitive data

We use process protection technology based on a VMM, through a trusted VMM layer, and bypass the guest operating system, providing data protection directly to the user process. To protect data security in the process of interaction on the cloud platform, the following steps must be completed. Establishing a credible environment and channels During the booting process, the cloud platform needs to measure startup software through trusted computing technology.

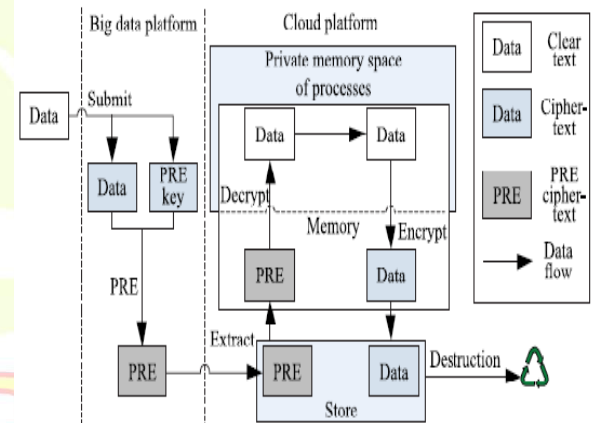


Fig. 4 Safe operation process.

The SESP must establish a reliable channel with the VMM in the cloud, and then receive sensitive data safely from the big data platform. The remote attestation and hand shaking protocol between the SESP and the VMM in the cloud is shown in Fig. 5. In fact, the VMM responds to the request at the cloud server end. First, the SESP sends an integrity request to the cloud server, including the SESP public key (PK_{id}) and timestamp (TS). Second, the VMM generates a session key (K_{sess}) and computes the hashed value of TS, PK_{id} , and K_{sess} using Secure Hash Algorithm (SHA1). Then, the VMM calls the TPM quote instruction and passes the hashed value and PCR as arguments to obtain the testimony (Quote) using the TPM private key signature. The VMM uses PK_{id} to encrypt K_{sess} and then sends K_{sess} , Quote, and a Certification Authority (CA) certificate to the other

side. The SESP verifies the value of TS, PK_{id} , and K_{sess} after receiving this information. If the values are consistent, the communications are secure. As a result, both sides of the communications determine a sessionkey. In the future, both sides of communication will be encrypted using the session key.

(2) Data upload and extraction

The cloud users (SESP) extract the sensitive data from the big data platform through retrieval. We assume that the cloud is untrusted. The uploaded executable application and data must be encrypted before the SESP uses the cloud. The upload and extract protocol of the data, the SESP generates the AES symmetric key and a pair of asymmetric keys (PK_{app} , SK_{app}) using the tools, encrypts the executable files and data files using the AES symmetric key, and encrypts the AES key by the asymmetric keys, which are attached at the end of the application files. The user encrypts the PK_{id} , registration command, application name, public key (PK_{app}), and predetermined lease using K_{sess} , and then sends them to the VMM. Finally, encrypted executable files and data files are uploaded to the cloud server.

(3) Program execution:

In the process of application execution on the cloud platform, dynamic data protection and encryption are similar to the protection of process memory space process execution, the Occupied memory process cannot be accessed by other processes and operating systems. The VMM serves as the bridge of data exchange between the operating system and the user process. When the OS copies the data from the user memory space, the VMM, not the operating system, performs the copying operation, because the operating system lacks read and write privileges. When the data are copied into the private memory space of the process, the VMM decrypts the data using the corresponding AES symmetric key. Thus, the data can be computed normally. Conversely, when the data in the private memory space of the process are copied to the outside, the VMM encrypts the data using the corresponding AES symmetric key.

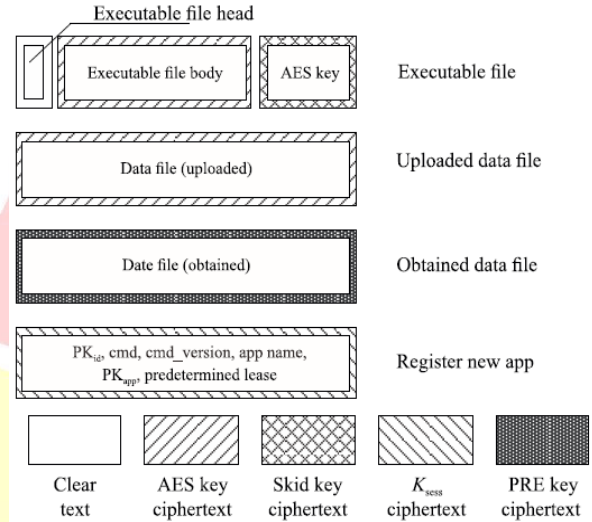


Fig. 6 Upload and extract protocol of the data.

Hence, the user data stored on disk is in ciphertext form. In a word, in the private space of a user process, the security plug-in decrypts PRE data from the big data platform and the VMM decrypts data from the cloud user (SESP). The generated data are encrypted when the user process is completed, and then the data is destroyed according to the terms of the lease. Therefore, the private space of the user process acts as a balance point of the security mechanism between the data owner and user, benefiting both while preventing sensitive leakage.

VI. CONCLUSION AND FUTURE WORKS

The main goal is to securely store, share and handle the data that is not forbidden by the owner of the data to provide security for data through homomorphic session based proxy re-encryption algorithm which are store up in the big data platform. In summary, we proposed a systematic framework of secure sharing of sensitive data on big data platform, which ensures secure submission and storage of sensitive data and guarantees secure use of clear text in the cloud platform by the private space of user process based on the VMM. The proposed framework well protects the security of users' sensitive data. At the same time the data owners have the complete control of their own data, which is a feasible solution to balance the benefits of involved parties under the semi-trusted conditions. In the future, we will optimize the heterogeneous proxy re-encryption algorithm, and further improve the efficiency of encryption. In addition, reducing the overhead of the interaction among involved parties is also an important future work.

REFERENCES

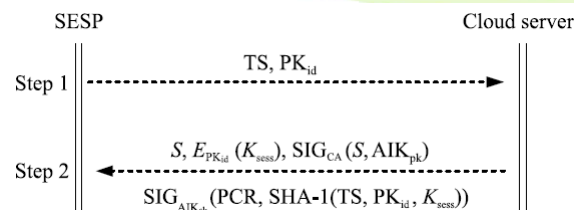



Fig. 5 Remote attestation and hand shaking protocol between the SESP and the VMM in the cloud.

- 
- [1] A. M. Azab, P. Ning, E. C. Sezer, and X. Zhang, HIMA: A hypervisor-based integrity measurement agent, in Proc. 25th Annual Computer Security Applications Conf., Hawaii, USA, 2009, pp. 461–470.
- [2] A. M. Azab, P. Ning, Z. Wang, X. Jiang, X. Zhang, and N. C. Skalsky, HyperSentry: Enabling stealthy in-context measurement of hypervisor integrity, in Proc. 17th ACM Conference on Computer and Communications Security, Chicago, USA, 2010, pp. 38–49.
- [3] Trusted Computing Group, TNC architecture for interoperability, <http://www.trustedcomputinggroup.org/resources/tnc-architecture-for-interoperability-specification>, 2014.
- [4] H. Zhang, L. Chen, and L. Zhang, Research on trusted network connection, (in Chinese), Chinese Journal of Computers, vol. 33, no. 4.
- [5] D. Feng, Y. Qin, D. Wang, and X. Chu, Research on trusted computing technology, (in Chinese), Journal of Computer Research and Development.
- [6] F. Zhang, J. Chen, H. Chen, and B. Zang, Cloudvisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization.
- [7] X. Chen, T. Garfinkel, E. C. Lewis, and B. Spasojevic, Overshadow: A virtualization-based approach to retrofitting protection in commodity operating systems.
- [8] J. Yang and K. G. Shin, Using hypervisor to provide data secrecy for user applications on a per-page basis, in Proc. 4th Int. Conf. on Virtual Execution Environments, Seattle, USA, 2008, pp. 71–80.