Tsinghua Science and Technology

Volume 20 | Issue 1 Article 8

2015

Secure Sensitive Data Sharing on a Big Data Platform

Xinhua Dong
the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

Ruixuan Li
the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

Heng He
the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

Wanwan Zhou
the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

Zhengyuan Xue
the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

See next page for additional authors

Recommended Citation

Xinhua Dong, Ruixuan Li, Heng He et al. Secure Sensitive Data Sharing on a Big Data Platform. Tsinghua Science and Technology 2015, 20(1): 72-80.

Secure Sensitive Data Sharing on a Big Data Platform

Authors

Xinhua Dong, Ruixuan Li, Heng He, Wanwan Zhou, Zhengyuan Xue, and Hao Wu

TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 08/11 pp72-80
Volume 20, Number 1, February 2015

Secure Sensitive Data Sharing on a Big Data Platform

Xinhua Dong, Ruixuan Li*, Heng He, Wanwan Zhou, Zhengyuan Xue, and Hao Wu

Abstract: Users store vast amounts of sensitive data on a big data platform. Sharing sensitive data will help enterprises reduce the cost of providing users with personalized services and provide value-added data services. However, secure data sharing is problematic. This paper proposes a framework for secure sensitive data sharing on a big data platform, including secure data delivery, storage, usage, and destruction on a semi-trusted big data sharing platform. We present a proxy re-encryption algorithm based on heterogeneous ciphertext transformation and a user process protection method based on a virtual machine monitor, which provides support for the realization of system functions. The framework protects the security of users' sensitive data effectively and shares these data safely. At the same time, data owners retain complete control of their own data in a sound environment for modern Internet information security.

Key words: secure sharing; sensitive data; big data; proxy re-encryption; private space

1 Introduction

With the rapid development of information digitization, massive amounts of structured, semi-structured, and unstructured data are generated quickly. By collecting, sorting, analyzing, and mining these data, an enterprise can obtain large amounts of individual users' sensitive data. These data not only meet the demands of the enterprise itself, but also provide services to other businesses if the data are stored on a big data platform. Traditional cloud storage merely stores plain text or encrypted data passively. Such data can be considered as "dead", because they are not involved in calculation. 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 (such as encrypting data, conversion, or function encryption) on data used by participants, which can invigorate "dead" data. An example of such an application is shown in Fig. 1 to illustrate the flow process of sensitive data on such a platform.

In Fig. 1, we consider user's preferences as sensitive data. When Alice submits a query (sportswear), 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, "badminton", then the search engine returns personalized results (sportswear + badminton) to Alice. When Alice sees

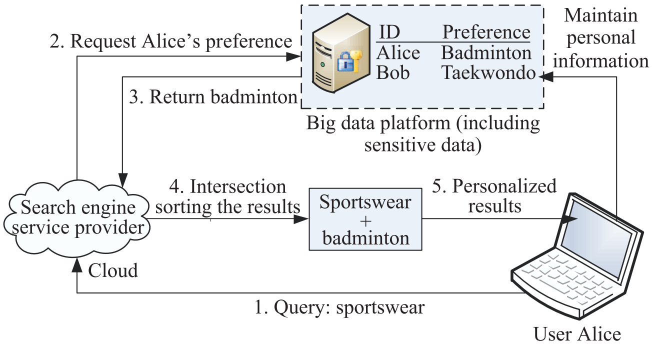
2. Request Alice's preference
ID
Preference
Alice
Badminton
Bob
Taekwondo
Maintain personal information
3. Return badminton
Big data platform (including sensitive data)
4. Intersection sorting the results
5. Personalized results
Search engine service provider
Sportswear + badminton
Cloud
1. Query: sportswear
User Alice

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

• Xinhua Dong, Ruixuan Li, Heng He, Wanwan Zhou, Zhengyuan Xue, and Hao Wu are with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China. E-mail: {xhdong, rxli}@hust.edu.cn; willam1981@gmail.com; {zhouwanwan, xue, hwu2013}@hust.edu.cn.

∗ To whom correspondence should be addressed.

Manuscript received: 2014-12-05; accepted: 2014-12-25

her favorite badminton sportswear, she experiences a pleasant purchase. Consequently, this leads to a win-win situation. However, while data sharing increases enterprise assets, Internet insecurity and the potential of sensitive data leakage also create security issues for sensitive data sharing.

Secure sensitive data sharing involves four primary safety factors. First, there are security issues when sensitive data are transmitted from a data owner's local server to a big data platform. Second, there can be sensitive data computing and storage security problems on the big data platform. Third, there are secure sensitive data use issues on the cloud platform. Fourth, there are issues involving secure data destruction. Some research institutions and scholars at home and abroad have made positive contributions to exploration and research aimed at solving these security problems.

Existing technologies have partially resolved data sharing and privacy protection issues from various perspectives, but they have not considered the entire process in the full data security life cycle. However, a big data platform is a complete system with multi-stakeholder involvement, and thus cannot tolerate any security breach resulting in sensitive data loss. In this paper, we analyze security issues involving the entire sensitive data sharing life cycle and describe a system model created to ensure secure sensitive data sharing on a big data platform, to guarantee secure storage on the big data platform using Proxy Re-Encryption (PRE) technology, and to ensure secure use of sensitive data sharing using a private space process based on a Virtual Machine Monitor (VMM). Then, a security plug-in and a data self-destruction mechanism help to alleviate user concern regarding sensitive personal information leakage.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 proposes a systematic framework for secure sensitive data sharing. Section 4 explains secure submission and storage of sensitive data based on PRE in detail. Section 5 provides our solution for ensuring secure sensitive data use based on a VMM. Conclusions are drawn in Section 6.

2 Related Work

In this section, we focus on previous work on relevant topics such as encryption, access control, trusted computing, and data security destruction technology in a cloud storage environment.

2.1 Data encryption and access control of cloud storage

Regarding encryption technology, the Attribute-Based Encryption (ABE) algorithm includes Key-Policy ABE (KP-ABE)[1] and Ciphertext-Policy ABE (CP-ABE)[2]. ABE decryption rules are contained in the encryption algorithm, avoiding the costs of frequent key distribution in ciphertext access control. However, when the access control strategy changes dynamically, a data owner is required to re-encrypt the data. A method based on PRE is proposed in Ref. [3]. 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[4]. A Fully Homomorphic Encryption (FHE) mechanism is proposed in Ref. [5]. 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 (AB-ACCS), 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 according to Shamir's theory of secret sharing. With a Single Sign-On (SSO), any authorized user can log in to the cloud storage system using a standard common application interface.

2.2 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. In academia, 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. Integrity measurement is the primary technical means of building a trusted terminal platform. Research on virtual platform measurement technology includes HIMA[12] and HyperSentry[13] metric architecture. Using virtual platform isolation features, HIMA measures the integrity of a virtual machine by monitoring the virtual machine's memory. HyperSentry completes the integrity measurement using a hardware mechanism. TCG issued a Trusted Network Connection (TNC) architecture specification version 1.0[14] in 2005, characterized by having terminal integrity as a decision of network access control. Chinese scholars have conducted research on trusted network connections based on the TNC architecture[15]. Beginning by establishing the trust of the terminal platform, 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 Xen VMM and a Confidentiality and High-Assurance Equipped Operating System (CHAOS)[18–21]. 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.

2.3 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. In Ref. [26], 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. In Ref. [27], Razick et al. 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.

3 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. 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 Heterogeneous Proxy Re-Encryption (H-PRE), which supports heterogeneous 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

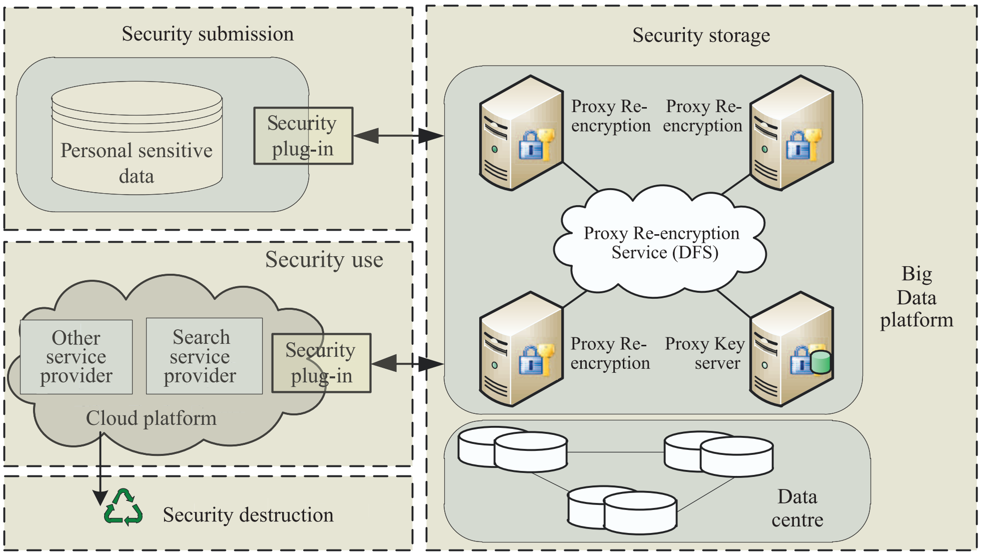

Fig. 2 Systematic framework for secure sensitive data sharing on a big data platform.

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. Cleartext 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 plug-in. 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.

4 Secure Submission and Storage of Sensitive Data Based on PRE

4.1 H-PRE

H-PRE involves three types of algorithm, traditional identity-based encryption (including $\text{Setup}_{\text{IBE}}$, $\text{KeyGen}_{\text{IBE}}$, Enc_{IBE}, and Dec_{IBE}), re-encryption (including $\text{KeyGen}_{\text{RE}}$, ReEnc, and ReDec functions), and the last one is the traditional public key cryptosystems (including $\text{KeyGen}_{\text{PKE}}$, Enc_{PKE}, and Dec_{PKE}). 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) $\text{Setup}_{\text{IBE}}(k)$: 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) $\text{KeyGen}_{\text{IBE}}(\text{mk, params, id})$: 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) $\text{KeyGen}_{\text{PKE}}(\text{params})$: 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) $\text{Enc}_{\text{IBE}}(\text{pk}_{\text{id}}, \text{sk}_{\text{id}}, \text{params}, m)$: When the user encrypts data, the data owner encrypts the clear-text (m) into the ciphertext ($c = (c_1, c_2)$) using the user's own (pk_{id}, sk_{id}) and a random number ($r \in \text{RZ}_p^*$).

(5) $\text{KeyGen}_{\text{RE}}(\text{sk}_{\text{id}_i}, \text{sk}'_{\text{id}_i}, \text{pk}'_{\text{id}_j}, \text{params})$: When the data owner (user i) grants user j permissions, using sk_{id_i}, sk'_{id_i}, and pk'_{id_j}, user i computes the PRE key ($\text{rk}_{\text{id}_i-\text{id}_j}$), completing the transformation from user i to user j.

(6) $\text{ReEnc}(c_i, \text{rk}_{\text{id}_i-\text{id}_j}, \text{params})$: 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 ($\text{rk}_{\text{id}_i-\text{id}_j}$), and related system parameters, and then the big data platform computes and outputs the PRE ciphertext ($c_j = (c_{j1}, c_{j2})$).

(7) $\text{Dec}_{\text{PKE}}(c_j, \text{sk}'_{\text{id}_j}, \text{params})$: 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 ($m' = m$) of the data using his or her own sk'_{id_j}.

4.2 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. Moreover, the computing resources of the big data platform are more powerful than those of the client. Hence, we can put PRE computational overhead on the big data platform to improve user experience. 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 DecPKE 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.

5 Secure Use of Sensitive Data on VMM

5.1 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 process is shown in Fig. 4.

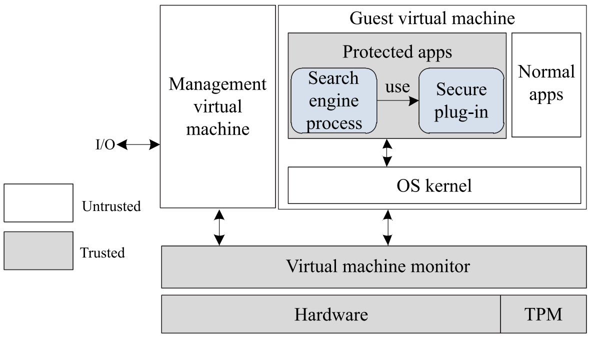

Fig. 3 Threat model of a sensitive process in a cloud platform.

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). 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.

5.2 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.

(1) Establishing a credible environment and channels

During the booting process, the cloud platform needs to measure startup software through trusted computing technology. Therefore, cloud users (SESPs) must ensure the integrity of the VMM, that is to say, cloud users must ensure that the VMM is trusted. After the booting process, the cloud server will store Basic Input/Output System (BIOS), GRand Unified Bootloader (GRUB), and VMM measurements in the Platform Configuration Register (PCR) of the TPM chip, and then send a remote verification to the user to ensure the trust relationship between them.

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

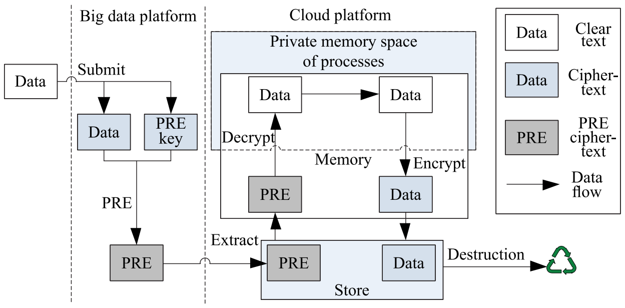

Fig. 4 Safe operation process.

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 integrality request to the cloud server, including the SESP public key ($\mathrm{PK_{id}}$) and timestamp (TS). Second, the VMM generates a session key (K_{sess}) and computes the hashed value of TS, $\mathrm{PK_{id}}$, 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 $\mathrm{PK_{id}}$ 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, $\mathrm{PK_{id}}$, 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 session key. 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 is shown in Fig. 6.

In Fig. 6, the SESP generates the AES symmetric key and a pair of asymmetric keys ($\mathrm{PK_{app}}$, $\mathrm{SK_{app}}$) 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 data obtained from the big data platform are PRE ciphertext, which can be decrypted during runtime. The command format of the new program must be identified when registering the program. The user encrypts the $\mathrm{PK_{id}}$, registration command, application name, public key ($\mathrm{PK_{app}}$), 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.

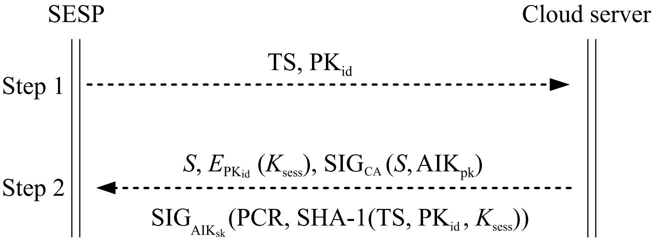
SESP
Cloud server
Step 1
TS, $\mathrm{PK_{id}}$
Step 2
S, $E_{\mathrm{PK_{id}}}(K_{\mathrm{sess}})$, $\mathrm{SIG_{CA}}(S, \mathrm{AIK_{pk}})$
$\mathrm{SIG_{AIK_{sk}}}(\mathrm{PCR}, \text{SHA-1}(\mathrm{TS}, \mathrm{PK_{id}}, K_{\mathrm{sess}}))$

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

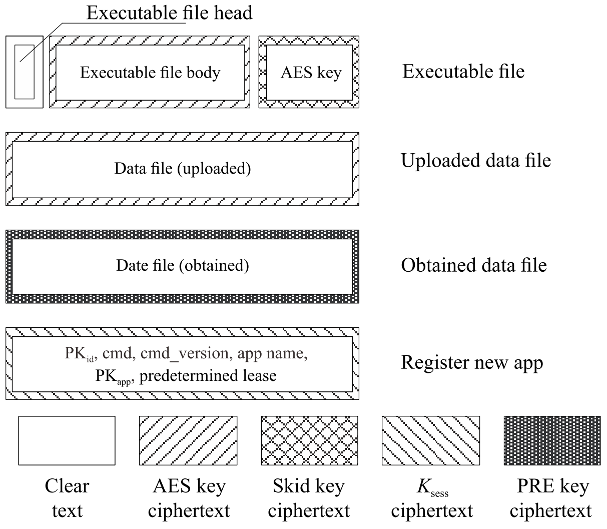

Fig. 6 Upload and extract protocol of the data.

(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[18–21], as shown in Fig. 4. During 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. 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 information leakage.

6 Conclusions

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 based on the heterogeneous proxy re-encryption algorithm, 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.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Nos. 61173170, 61300222, 61433006, and U1401258), and Independent Innovation Fund of Huazhong University of Science and Technology (Nos. 2012TS052, 2012TS053, 2013QN120, and CXY13Q019). We sincerely thank the anonymous reviewers for their very comprehensive and constructive comments.

References

[1] S. Yu, C. Wang, K. Ren, and W. Lou, Attribute based data sharing with attribute revocation, in *Proc. 5th ACM Symposium on Information, Computer and Communications Security*, Beijing, China, 2010, pp. 261–270.

[2] J. Bethencourt, A. Sahai, and B. Waters, Ciphertext-policy attribute-based encryption, in *Proc. IEEE Symposium on Security and Privacy*, Oakland, USA, 2007, pp. 321–334.

[3] J. Li, G. Zhao, X. Chen, D. Xie, C. Rong, W. Li, L. Tang, and Y. Tang, Fine-grained data access control systems with user accountability in cloud computing, in *Proc. 2nd Int. Conf. on Cloud Computing*, Indianapolis, USA, 2010, pp. 89–96.

[4] L. Wang, L. Wang, M. Mambo, and E. Okamoto, New identity-based proxy re-encryption schemes to prevent collusion attacks, in *Proc. 4th Int. Conf. Pairing-Based Cryptograghy-Pairing*, Ishikawa, Japan, 2010, pp. 327–346.

[5] C. Gentry, A fully homorphic encryption scheme, Ph.D dissertation, Stanford University, California, USA, 2009.

[6] S. Ananthi, M.S. Sendil, and S. Karthik, Privacy preserving keyword search over encrypted cloud data, in *Proc. 1st Advances in Computing and Communications*, Kochi, India, 2011, pp. 480–487.

[7] H. Hu, J. Xu, C. Ren, and B. Choi, Processing private queries over untrusted data cloud through privacy homomorphism, in *Proc. 27th IEEE Int. Conf. on Data Engineering*, Hannover, Germany, 2011, pp. 601–612.

[8] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, Privacy-preserving multi-keyword ranked search over encrypted cloud data, in *Proc. 30th IEEE INFOCOM*, Shanghai, China, 2011, pp. 829–837.

[9] C. Hong, M. Zhang, and D. Feng, AB-ACCS: A cryptographic access control scheme for cloud storage, (in Chinese), *Journal of Computer Research and Development,* vol. 47, no. 1, pp. 259–265, 2010.

[10] N. Zeldovich, S. Boyd-Wickizer, and D. Mazieres, Securing distributed systems with information flow control, in *Proc. 5th USENIX Symposium on Networked Systems Design and Implementation*, San Francisco, USA, 2008, pp. 293–308.

[11] Z. Lv, C. Hong, M. Zhang, and D. Feng, A secure and efficient revocation scheme for fine-grained access control in cloud storage, in *Proc. 4th IEEE Int. Conf. on Cloud Computing Technology and Science*, Taipei, Taiwan, China, 2012, pp. 545–550.

[12] 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.

[13] 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.

[14] Trusted Computing Group, TNC architecture for interoperability, http://www.trustedcomputinggroup.org/resources/tnc_architecture_for_interoperability_specification, 2014.

[15] H. Zhang, L. Chen, and L. Zhang, Research on trusted network connection, (in Chinese), *Chinese Journal of Computers,* vol. 33, no. 4, pp. 706–717, 2010.

[16] D. Feng, Y. Qin, D. Wang, and X. Chu, Research on trusted computing technology, (in Chinese), *Journal of Computer Research and Development,* vol. 48, no. 8, pp. 1332–1349, 2011.

[17] F. Zhang, J. Chen, H. Chen, and B. Zang, Cloudvisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization, in *Proc. 23rd ACM Symposium on Operating Systems Principles*, Cascais, Portugal, 2011, pp. 203–216.

[18] X. Chen, T. Garfinkel, E. C. Lewis, and B. Spasojevic, Overshadow: A virtualization-based approach to retrofitting protection in commodity operating systems, in *Proc. 13th Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, Seattle, USA, 2008, pp. 2–13.

[19] 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.

[20] H. Chen, F. Zhang, C. Chen, Z. Yang, R. Chen, B. Zang, W. Mao, H. Chen, F. Zhang, C. Chen, et al., Tamper-resistant execution in an untrusted operating system using a virtual machine monitor, Technical Report, Parallel Processing Institute, Fudan University, FDUPPITR-2007-0801, 2007.

[21] P. Dewan, D. Durham, H. Khosravi, M. Long, and G. Nagabhushan, A hypervisor-based system for protecting software runtime memory and persistent storage, in *Proc. the 2008 Spring Simulation Multiconference*, Ottawa, Canada, 2008, pp. 828–835.

[22] G. Wang, F. Yue, and Q. Liu, A secure self-destructing scheme for electronic data, *Journal of Computer and System Sciences,* vol. 79, no. 2, pp. 279–290, 2013.

[23] L. Zeng, Z. Shi, S. Xu, and D. Feng, Safevanish: An improved data self-destruction for protecting data privacy, in *Proc. 2nd Cloud Computing International Conf.*, Indianapolis, USA, 2010, pp. 521–528.

[24] L. Dong, Y. Zhuang, Y. Gao, and Y. Bu, Research on real-time trigger system for sensitive data safe destruction, (in Chinese), *Journal of Chinese Computer System*, vol. 31, no. 7, pp. 1323–1327, 2010.

[25] J. Qin, Q. Deng, and J. Zhang, Design of multi-grade safety data destruction mechanism of HDFS, (in Chinese), *Computer Technology and Development*, vol. 23, no. 3, pp. 129–133, 2013.

[26] F. Zhang, J. Chen, H. Chen, and B. Zang, Lifetime privacy and self-destruction of data in the cloud, (in Chinese), *Journal of Computer Research and Development*, vol. 48, no. 7, pp. 1155–1167, 2011.

[27] S. Razick, R. Mocnik, L. F. Thomas, E. Ryeng, F. Drabløs, and P. Sætrom, The eGenVar data management system — Cataloguing and sharing sensitive data and metadata for the life sciences, *Database,* vol. 2014, p. bau027, 2014.

Xinhua Dong is a PhD student in the School of Computer Science and Technology, Huazhong University of Science and Technology, Whuan, China. He received his MS degree from Huazhong University of Science and Technology in 2008. His research interests include information retrieval, cloud security, and big data management. He is a student member of the CCF.

Ruixuan Li is currently a professor in the School of Computer Science and Technology, HUST, and an adjunct associate professor at Concordia University, Canada. He received his BS, MS, and PhD degrees from Huazhong University of Science of Science and Technology in 1997, 2000, and 2004, respectively. From 2009–2010, he was a visiting researcher at the University of Toronto. His research interests include cloud computing, big data management, social networking, and distributed system security. He is a member of IEEE and ACM and a senior member of the CCF.

Heng He is a PhD student in the School of Computer Science and Technology, Huazhong University of Science and Technology, Whuan, China. He received his MS degree from Huazhong University of Science and Technology in 2007. His research interests include cloud computing, distributed simulation, and network security.

Wanwan Zhou is a master student in the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. She received her BS degree from Huazhong University of Science and Technology in 2012. Her research interests include information retrieval, cloud computing, and big data security.

Zhengyuan Xue is a PhD student in the School of Computer Science and Technology, Huazhong University of Science and Technology, Whuan, China. He received his MS degree from Huazhong University of Science and Technology in 2012. His research interests include information retrieval, cloud security, and big data management.

Hao Wu is a master student in the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China. He received his BS degree from Tianjin Polytechnic University, China in 2013. His research interests include information retrieval, cloud computing, and big data security.