Advances in Computer Science Research, volume 65

3rd International Conference on Communications, Information Management and Network Security (CIMNS 2018)

Design of the Core Module of a Key Management Server

Jianyu Ai [1], Hao Huang [1, 2, *] and Lei Shi[1]

[1] School of Software Engineering, HuaZhong University of Science & Technology, Wuhan, China

[2] Shenzhen Research Institute, Huazhong University of Science & Technology, Shenzhen, China

[*]Corresponding author

***Abstract*—This paper researches the key management server system to solve the problems of distribution, storage and backup of massive key management under the content association key encryption mechanism, and provides technical support for the application of content association key technology.**

The core function of the key management server is to manage the key and process the related request of the key. The paper focuses on the design and implementation of the key management module. The key management module is responsible for processing the related requests of the user key. The module includes functions such as key uploading and downloading, key replacement, and key revocation. The test results show that the key management module can undertake the task of key management for the key management server system based on the content association key.

Keywords—content association key; key management server; key upload and download

I. INTRODUCTION

With the development of information technology, people pay more and more attention to the security of big data. When using cryptographic techniques to ensure data security, in terms of traditional encryption mechanisms, in the context of big data, both symmetric key systems and asymmetric key systems face a dilemma in computational efficiency and security.

The content-associated key technology is a new encryption mechanism. The core idea is to divide the plaintext data to be encrypted into a small amount of sensitive data with high importance and the main data with low importance, and extract the sensitive data as a key. The remaining subject data is treated as ciphertext after certain processing, and the ciphertext cannot express the original plaintext data application information because of lack of important data parts. Content-associated keying technology has the following advantages over traditional encryption mechanisms:

(1) One file and one key, the key data has a unique correspondence with the ciphertext due to the plaintext data to be encrypted;

(2) The number of key bits can reach KB level or even MB, and the corresponding calculation amount is much smaller than the traditional encryption mechanism.

A large number of data files can be encrypted by using content-associated key technology. However, as the number of users and the number of encrypted files continue to increase, how to manage massive keys becomes a new problem. Users cannot manually manage a large number of KB-byte keys. In addition, users may lack professional key management methods on the client side, and it is difficult to secure the keys.

For the above drawbacks, it is a good solution to establish a special key management server and manage the massive key with the software system as the support. The key management server centralizes all the users' keys and organizes and manages the keys uniformly. This avoids the security problems caused by users managing the keys by themselves. This paper will design and implement a key management server based on content association key technology.

II. KEY MANAGEMENT SERVER SYSTEM

The key management server based on the content association key encryption system should have the following basic functions:

(1) Identity authentication. All users of the system use the username/password for identity authentication.

(2) User information management function. The user can call the relevant interface to manage personal information in the system.

(3) Key management function. Include key upload and download, key revocation, and key replacement. After the user is authenticated, the user can call the server-related interface to upload the key to the server, and can download or revoke the key within the scope of his own authority. At the same time, the user can upload a new key to overwrite the historical key.

(4) Key security and storage function. Include key secondary encryption and key storage and key backup.

(5) Log auditing function. Support unified logging /query /audit user operational behavior and system events.

The core function of the key management server is to manage the key and process the related request of the key. The system involves mutual calling between multiple modules when processing the key request. Due to space limitations, this paper takes the key management module as an example to develop the design.

III. KEY MANAGEMENT MODULE DESIGN

The key management module is responsible for processing user key related requests. The module includes functions such as key uploading and downloading, key replacement, and key revocation. As the core resource managed by the system, the key must ensure that users with access rights can access it normally.

A. Key Upload Process

After receiving the key upload request from the client, the key management module first queries the database to determine whether the user has uploaded the key according to the user information transmitted from the communication routing module and the key name in the request, and if so, returns and prompts the user key already exists, otherwise receive the key data uploaded by the user.

After receiving the key data, first generate a unique number for the key that acts inside the system, and save the number and the correspondence information between the user and the key to the database. Then, the response client key upload is successful. At the same time, the system sends a key upload log report to the security audit module for archiving.

In general, keys are stored in addition to authentication to ensure confidentiality and integrity to prevent leakage and tampering. Therefore, the key needs to be encrypted so that the key is stored in the form of ciphertext.

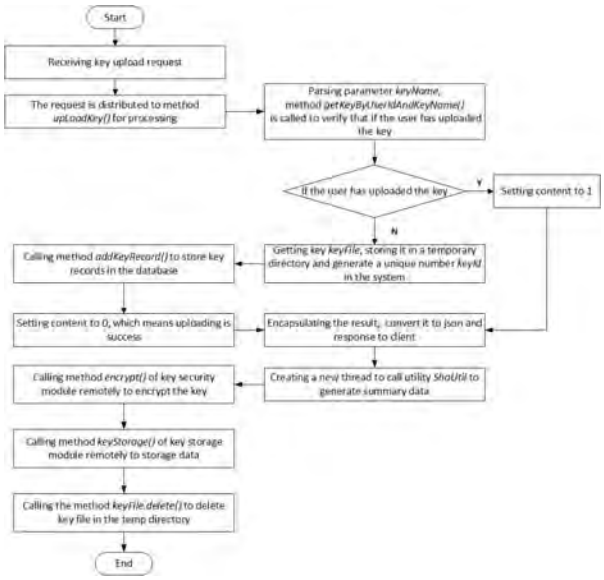

FIGURE I. KEY UPLOAD IMPLEMENTATION PROCESS

B. Key Download Process

In the system, users can download keys within their own scope of authority.

If the user has the right, the export interface of the key storage module is remotely called to obtain the corresponding key. Since the key is stored in the ciphertext form, after receiving the returned second ciphertext, the key management module first strips out the digest data, and then remotely calls the key security module to decrypt it.

After receiving the key data in the plaintext state returned by the key security module, the new digest data is calculated for the key data by using the same signature algorithm. Compare the two pieces of summary data. If they are inconsistent, call the key storage interface of the key storage module and pass in the specified key number to obtain the key in the backup library.

If they are consistent, the key is returned to the client by the secure channel. Simultaneously send a key download log report to the security audit module for archiving. At this point, the process of a key download ends.

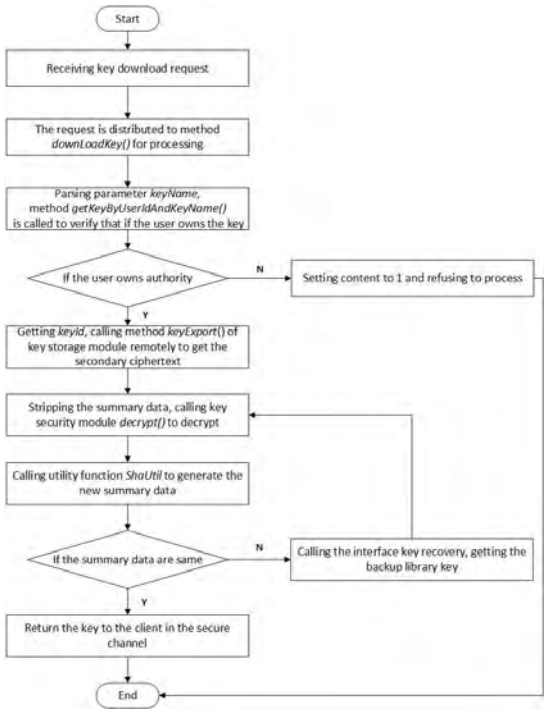

FIGURE II. KEY DOWNLOAD IMPLEMENTATION PROCESS

C. Key Replacement Process

Under the content-related key encryption system, in the case of the same plaintext data, when the security strength requirements are different, the algorithm adjusts the encryption parameters so that the keys generated after extracting the encryption are different. Therefore, when the user re-encrypts the same plaintext data, the generated new key can be uploaded to the system to replace the uploaded historical key. For historical keys that need to be replaced, the system does not delete them, but instead keeps the historical key for a recent fixed number of versions in chronological order.

After receiving the user key replacement request forwarded by the communication routing module, the key management module first performs rights authentication on the user, and according to the user information and the key information specified in the request, queries the database to determine

whether the user has the key. If the key does not exist or does not have permission, it will directly refuse processing.

After the authority is authenticated, the information such as the upload time of the corresponding key in the database is updated, and the rest of the information is consistent. At this time, after receiving the new key uploaded by the client, the response to the client key replacement process is successful. Simultaneously send a key replacement log report to the security audit module for archiving.

After receiving the new key uploaded by the client, the key management module will call the signature algorithm and the key security module to perform security processing in the background, and finally call the update interface of the key storage module to perform the key update operation.

D. Key Revocation Process

The user can undo the uploaded key. As with the changed history key, the system does not delete it, but retains the last version of the key for it.

After the key management module receives the key revocation request, it is consistent with the key replacement. First, the user is authenticated by the query, and the query database is used to determine whether the user has key ownership. If it has, the delete interface provided by the key storage module is called, and the corresponding number of the key is transmitted to the module to perform a key deletion operation.

Finally, the client key revocation process is successfully processed, and a key revocation log report is sent to the security audit module for record archiving.

IV. SYSTEM FUNCTION TEST

(1) First construct and send a login request. After calling the server-side verification code interface to obtain the verification code, type the correct user's mobile phone number, password, and verification code. The content-type is set to application/json, the request mode is set to post, and the server IP is directed to the communication routing module.

(2) After the client successfully receives the response, it constructs and sends a key upload request. The key generated by the content association key encryption algorithm is selected, the key size is 5168 bytes, and the key name keyName is set to 201802241037887217. The content-type is set to multipart/form-data, and the request is sent by the post method. The server IP also points to the communication routing module. At the same time, the returned token is appended to the request header. As shown in Figure 3.

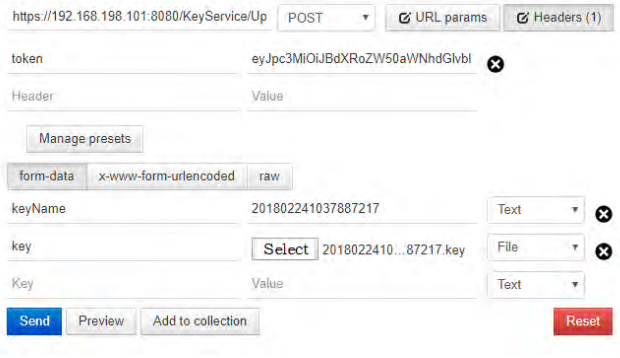

FIGURE III. KEY UPLOAD REQUEST

(3) After the communication routing module authenticates the token, it routes it to the key management module for processing. Since the key was not previously uploaded, the key management module receives the key and generates a unique number for it inside the system, and saves the key information to the database.

As shown in Figure 4, according to the key name KeyName and the user mobile phone number UserPhone, the information record of the corresponding key in the database can be queried in the keyinfo table. The unique number generated by the system for this key is d5283042be7d4a3a961f797a2b514cd0.

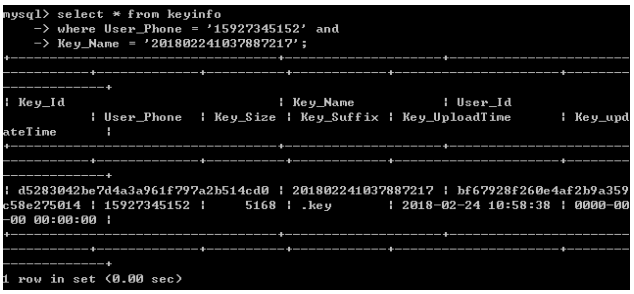

FIGURE IV. KEY INFORMATION RECORD UNDER MYSQL COMMAND LINE

(4) The client can correctly receive the response. The HTTP status code 200 indicates that the request response is successful. The status value of 0 indicates that the server has successfully processed; the content value of 0 indicates that the key is successfully uploaded. As shown in Figure 5.

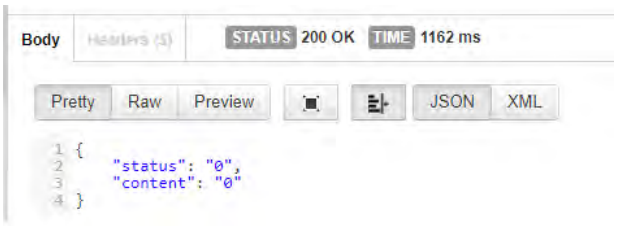

FIGURE V. KEY UPLOAD SUCCEEDED

The test results show that after the user obtains the token with the correct username and password, the content association key can be successfully uploaded from the secure channel to the server for management. The key management

server saves the key related information and encrypts it to storage. When the user downloads the key, the server successfully decrypts and restores the key file, and returns the key to the client.

V. SUMMARY

This paper researches a key management server that handles the organization and management of massive keys generated by the content-related key encryption mechanism, and designs and implements its core functional module: key management module. Tests show that the module works with other user management modules to perform key upload, download, replacement, and undo functions.

ACKNOWLEDGMENTS

This work is co-funded by JCYJ20160531194457572, basic research project of Shenzhen science and Technology Innovation Committee, and 2015 R&D support foundation of Shenzhen Virtual University Park: Shenzhen branch of DSSL, project of research and platform construction, and in part by Independent innovation research foundation of HUST.

REFERENCES

[1] Feng Dengguo, Zhang Min, Li Wei. Big data security and privacy protection. Chinese Journal of Computers, 2014, 37(1):246-258

[2] Zhang Yong, Research on Some Problems of Key Management, [PhD thesis]. Shanghai: East China Normal University, 2013

[3] Srivatsa M, Iyengar A, Yin J, et al. Scalable key management algorithms for location-based services. IEEE/ACM Transactions on Networking, 2009, 17(5):1399-1412

[4] Wang Wei. Research on video copyright protection technology based on content association key: [PhD thesis]. Wuhan: Huazhong University of Science and Technology, 2015

[5] Ren Xinyu, Li Yi, Huang Jintao, et al. Design and Implementation of Encrypted Cloud Storage Security Framework Based on Key Policy Attributes. Communications Technology, 2017, 50(2): 340-345

[6] Zheng X, Huang C T, Matthews M. Chinese remainder theorem based group key management in: ACMSoutheast Regional Conference.Winston-Salem, NC, USA: ACM Association Press, 2007. 266-271

[7] Mossa E. Security Enhancement for AES Encrypted speech in communications. International Journal of Speech Technology, 2017, 20(1): 163-169

[8] Xu L, Jiang C, Wang J, et al. Information Security in Big Data: Privacy and Data Mining. IEEE Access, 2014, 2(2):1149-1176