A File cloud sharing method based on XOR operation

Yue He¹, Gao Xia² and Chengwan He¹,*

¹School of Information Engineering, Wuhan University of Technology, Wuhan, China
²School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China

*Corresponding author e-mail: hechengwan@hotmail.com

Abstract. To realize the secure storage and sharing of files, a file cloud storage method based on XOR (Exclusive OR) operation is proposed. In this method, a file is encrypted into multiple file slices based on XOR operation, then these file slices are uploaded to different locations of the public cloud platform. These file slices can be downloaded from the cloud platform and restored to the original data file based on XOR operation. In this way, even if a file slice is stolen, the information of the file will not be stolen. This paper gives the XOR-based encryption and restoration algorithms for the two cases where the original data file is not divided or divided into arbitrary secrets. Finally, we give the implementation method of the prototype.

1. Introduction

Since Shamir [1] published the paper "How to share a secret" in 1979, the (k, n) threshold secret sharing algorithm [2][3] has been widely used and developed. The (k, n) threshold algorithm refers to dispersing a secret S into n secret information. Any k secret information can recover the original secret S, but any k-1 secret information cannot recover S. The emergence of secret sharing solves the basic problem of secure key storage, which can not only make sure the security and integrity of secrets, but also prevent the risks caused by excessive concentration of secrets [3].

The traditional threshold implementation scheme has the problem of slow operation speed [4], XOR based secret sharing scheme has become an important research topic [4][5][6].

This paper proposes a simple and efficient file cloud sharing method based on XOR operation. It performs multiple XOR operations on the original data file and the randomly generated encryption keys, and then saves some of the ciphertext data blocks to the cloud platform. If necessary, download the corresponding data blocks from the cloud platform and restore them based on XOR operation. This paper gives the XOR-based encryption and restoration algorithms for the two cases where the original data file is not divided or divided into arbitrary secrets.

The rest of the paper is organized as follows: Section 2 describes the file encryption and decryption method based on XOR operation when the file is not split. Section 3 introduces the encryption and decryption method based on XOR operation when the file is divided into arbitrary secrets. Our implementation method is presented in Section 4 and Section 5 concludes the paper.
2. Encryption and decryption based on XOR operation

Instead of splitting the original data file, do the multiple XOR operations shown in Figure 1 to get the $n$ secrets. Where, Binary File represents the original file to be encrypted, Random Key 1..$n$ represent the randomly generated binary encryption keys, and Encrypted File 1..$n$ represent the encrypted data files.

If you want to save two data blocks on the cloud platform, do one encryption operation, and save Encrypted File 1 and Random Key 1. If you need to save three data blocks, you need to perform two encryption operations, and save Encrypted File 2, Random Key 2 and Random Key 1, and so on. In short, it is necessary to save the last encrypted data block and all the used encryption keys.

![Figure 1. File encryption process based on XOR operation](image)

The data represented by the dotted box in Figure 1 need to be saved on the cloud platform. The encryption algorithm is shown in Figure 2.

![Figure 2. XOR operation on a file](image)

The file restoration process is the opposite of the encryption process, as shown in Figure 3. Find the data file and the last used key file from all downloaded secrets, and do an XOR operation to get an intermediate file. Perform the XOR operation on this intermediate file and the next to the last key to get a new intermediate file. Repeat this process until the first key file is used for XOR operation to restore the original data file.
3. Encryption and decryption of multiple secrets

To further improve security, the original data file can be divided into $S_1$, $S_2$, ..., $S_n$ blocks, which are encrypted using XOR operations and stored on the cloud platform.

The following example shows the process of encryption and decryption based on XOR operation by dividing the original data file into 3 blocks. Where, $S_1$, $S_2$, and $S_3$ represent blocks of original files, $K_1$, $K_2$, and $K_3$ represent randomly generated encryption keys, $R_1$, $R_2$ and $R_3$ represent encryption results of $S_1$, $S_2$ and $S_3$, and $T_2$, $T_3$ represent intermediate data blocks to be saved.

$$
S_1 \oplus K_1 = R_1 \\
S_2 \oplus K_2 = R_2 \\
S_3 \oplus K_3 = R_3 \\
S_1 \oplus K_2 = T_2 \\
S_2 \oplus K_3 = T_3
$$

To achieve the restoration of the original data file, the data we need to save on the cloud platform include $R_1$, $R_2$, $R_3$, $T_1$, $T_2$, $K_1$. The process of data restoration is shown below.

$$
K_1 \oplus R_1 = S_1 \\
S_1 \oplus T_2 = K_2 \\
K_2 \oplus R_2 = S_2 \\
S_2 \oplus T_3 = K_3 \\
K_3 \oplus R_3 = S_3
$$

From the above process, it can be seen that the number of data blocks stored on the cloud platform should be $2 \times n$, where $n$ is the number of divided blocks of the original data file. See section 4 for details on how these data blocks are stored on the cloud platform.

The algorithm for encrypting multiple secrets of a file is shown in Figure 5.
Figure 5. The algorithm for encrypting multiple secrets of a file

The algorithm for restoring multiple secrets of a file is shown in Figure 6.

Figure 6. The algorithm for restoring multiple secrets of a file

4. Implementation

We have implemented a prototype system for file secret sharing. The architecture is shown in Figure 7. The execution flow of the program is as follows:

① The original data file is split into n secrets based on XOR operation;
② n secrets are uploaded to different buckets on the cloud platform;
③ Use email to notify other users of the download URLs;
④ Download n secrets;
⑤ Restore the original file based on the XOR operation.
The structure of each secret is shown in Figure 8. We put two data blocks in a secret. Each secret has a header containing two bytes, which holds the order of each secret and the size of the data block.

![Diagram of a secret]

Figure 8. The structure of a secret

For example, the original data file is divided into three secrets, whose structure and restoration process are shown in Figure 9.

![Diagram of secret storage and restoration]

Figure 9. Secret storage and restoration

5. Conclusion

This paper proposes a simple and efficient file cloud sharing method based on XOR operation. It performs multiple XOR operations on the original data file and the randomly generated encryption keys and then saves some of the ciphertext data blocks to the cloud platform. If necessary, download the corresponding data block from the cloud platform and restore it based on the XOR operation.

The method proposed in this paper can segment the original data file to any size, and can effectively manage the keys and reduce the time of data encryption and decryption. We will continue to improve the algorithm proposed in this paper to support the (k, n) threshold.

References

[1] Adi Shamir, How to share a secret, Communications of the ACM, 22(1979) 11, pp. 612-613.
[2] Blakley G., Safeguarding cryptographic key, Proceedings of AFIPS 1979 National Computer Conference, New York, USA, 1979, pp. 313-317.
[3] RONG Hui-gui, MO Jin-xia, CHANG Bing-guo, et al., Key distribution and recovery algorithm based on Shamir’s secret sharing. Journal on Communications, 36(2015)3, pp. 1-10.
[4] Masakatsu Matsuo, Kouji Mutou, (k,n)-Threshold Secret Sharing Scheme Using Exclusive OR, Panasonic Technical Journal, 59(2013) 2, pp.115-119.
[5] Binu, V. P., Sreekumar, A., Simple and Efficient Secret Sharing Schemes for Sharing Data and Image, International Journal of Computer Science and Information Technologies, 6(2015)1, pp.404-409.
[6] Lei Zhang and Douglas M. Blough, Deceptive Secret Sharing, 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2018, pp.442-453.