Abstract:
Deduplication techniques are widely employed to backup data and minimize network and storage overhead by detecting and eliminating redundancy among data. Instead of keeping multiple data copies with the same content, deduplication eliminates redundant data by keeping only one physical copy and referring other redundant data to that copy. Each such copy can be defined based on different granularities: it may refer to either a whole file (i.e., file level deduplication), or data block (i.e., block-level deduplication). To applying deduplication to user data to save maintenance cost in HDFS. Apart from normal encryption and decryption process we have proposed Master key concept with Dekey concept. For Encryption and Decryption, we have used Triple Data Encryption Standard Algorithm where the plain text is Encrypted triple times with the key so that the data is secure and reliable from hackers. We reduced the cost and time in uploading and downloading with storage space in HDFS. Deduplication can properly address the reliability and tag consistency problem in HDFS storage systems.

Keywords: Big Data, Cloud computing, dead ringer, dekey, HDFS, Message digest, SHA, Triple DES.

I. INTRODUCTION

Cloud computing is a type of internet-based computing that provides shared computer processing resources. The cloud aims to cut costs and help the users focus on their core business. It has adorable properties, such as scalability, elasticity, fault tolerance and pay-per-use. Thus, it has roseate service platform. Cloud provides an important service called data storage service. The term cloud service is a broad category that encompasses the myriad IT resources provided over the internet. With the potentially infinite storage space offered by cloud providers, users tend to use as much space as they can and vendors constantly look for techniques aimed to minimize redundant data and maximize space savings. Cloud users upload personal or confidential data to the data center of a cloud service provider(CSP) and allow it to maintain these data. Since intrusions and attacks towards sensitive data at CSP are not avoidable, so CSP cannot be fully trusted by cloud users. Hence, a good practice is to only outsource encrypted data to the cloud in order to ensure data security and user privacy. But the same or different users may upload duplicated data in encrypted form to CSP. Deduplication has proved to achieve high cost savings. Deduplication is an important technique to save the storage cost at the cloud storage server. Image is an important data type stored in cloud, but rarely discussed in previous work on deduplication. Deduplication can take place at either the file level or the block level. For file-level deduplication, it eliminates duplicate copies of the same file. Deduplication can also take place at the block level, which eliminates duplicate blocks of data that occur in non-identical files [18]. Specifically, traditional encryption requires different users to encrypt their data with their own keys. Thus, identical data copies of different users will lead to different cipher texts, making deduplication impossible. Convergent encryption [8] has been proposed to enforce data confidentiality while making deduplication feasible. It encrypts/decrypts a data copy with a convergent key, which is obtained by computing the cryptographic hash value of the content of the data copy. After key generation and data encryption, users retain the keys and send the cipher text to the cloud [18]. Existing solutions for deduplication suffer from brute force attacks [7],[11],[12],[13],[14]. They cannot flexibly support data access control and revocation at the same time. Most existing solutions cannot ensure reliability, security and privacy with sound performance [23],[24],[25]. In practice, it is hard to allow data holders to manage dead ringer avoidance due to a number of reasons. First, data holders may not be always online or available, which cause storage delay. Second, avoiding dead ringer could become to complicated in terms of communications and computations to involve data holders into deduplication process. Third, it may intrude privacy of data holders in the process of discovering duplicated data. Forth, the data holders have no idea of how to issue data access rights or deduplication keys. Therefore, CSP cannot cooperate with data holders on data storage deduplication in many situations. In this paper, we propose a scheme based on data ownership challenge and key management technique using DEKEY concept. We aim to solve the issue of dead ringer avoidance in the situation where multiple users upload the same data. Meanwhile, the performance of data deduplication in our scheme is not influenced by the size of the data, thus applicable for big data. Specifically, the contribution of this paper can be summarized as below:

- We propose an effective approach to verify the data ownership and check duplicate storage with secure challenge and big data support.

- Our scheme can flexibly support key sharing so even if the master key has lost we can recover our file by using shared keys.

- We prove the security and assess the performance of the proposed scheme through analysis and simulation. This result shows it efficiency, effectiveness and applicability.
II. METHODS AND MATERIALS

Data deduplication

As the volume of data stored in the cloud increases quickly, the term deduplication technique has more and more concerned recently. The strategies of deduplication can be categorized to two strategies: file-level and block-level deduplication. The file-level deduplication eliminates duplicate data copy at the file granularity if two files have the same hash value and are identified as identical [22][10][11]. This method needs low computational overhead but has low duplicate elimination effectiveness. The other block-level deduplication is also a popular technique which first divides each input file into several blocks of fixed-size or variable size and then use hash value of each block to eliminates the block already stored in cloud. Typical block size is 4KB to 8KB. Different data de-duplication products use different methods of breaking up the data into elements or chunks or blocks, but each product uses some technique to create a signature or identifier or fingerprint for each data element. As shown in the below figure, the data store contains the three unique data elements A, B, and C with a distinct signature. These data element signature values are compared to identify duplicate data. After the duplicate data is identified, one copy of each element is retained, pointers are created for the duplicate items, and the duplicate items are not stored. The basic concepts of data de-duplication are illustrated in below.

![Figure.1. Two methods frequently used for de-duplicating data are hash based and content aware.](image)

2.1 – Hash based Deduplication

Hash based data de-duplication methods use a hashing algorithm to identify “chunks” of data. Commonly used algorithms are Secure Hash Algorithm 1 (SHA-1) and Message-Digest Algorithm 5 (MD5). When data is processed by a hashing algorithm, a hash is created that represents the data. A hash is a bit string (128 bits for MD5 and 160 bits for SHA-1) that represents the data processed. If you process the same data through the hashing algorithm multiple times, the same hash is created each time. Hash based de-duplication breaks data into “chunks”, either fixed or variable length, and processes the “chunk” with the hashing algorithm to create a hash. If the hash already exists, the data is deemed to be a duplicate and is not stored. If the hash does not exist, then the data is stored and the hash index is updated with the new hash. In Figure 1-2, data “chunks” A, B, C, D, and E are processed by the hash algorithm and creates hashes Ah, Bh, Ch, Dh, and Eh; for purposes of this example, we assume this is all new data. Later, “chunks” A, B, C, D, and F are processed. F generates a new hash Fh. Since A, B, C, and D generated the same hash, the data is presumed to be the same data, so it is not stored again. Since F generates a new hash, the new hash and new data are stored.

![Figure.2. Hash Based de-duplication](image)

2.1.1 A fixed length or fixed block: In this data deduplication algorithm, it breaks the Data in to chunks or block, and the block size or block boundaries is Fixed like 4KB, or 8KB etc. And the block size never changes. While different devices/solutions may use different block sizes, the block size for a given device/solution using this method remains constant. The device/solution always calculates a hash value or signature on a fixed block and sees if there is a match. After a block is processed, it advances by exactly the same size and take another block and the process repeats. Advantage of dead ringer avoidance is it requires the minimum CPU overhead, fast and simple. It may be a drawback as the block size or block boundaries is Fixed, the main limitation of this approach is that when the data inside a file is shifted, for example, when adding a slide to a Microsoft PowerPoint deck, all subsequent blocks in the file will be rewritten and are likely to be considered as different from those in the original file. Smaller block size gives better deduplication than large ones, but it takes more processing to de duplicate. Larger block size gives low deduplication, but it takes less processing to de duplicate.

![Figure.3. So the Bottom line is Less storage savings and not efficient.](image)

B – Variable-Length or Variable Block

In this data deduplication algorithm, it breaks the Data in to chunks or block, and the block size or block boundaries is variable like 4KB, or 8KB or 16KB etc. And the block size changes dynamically during the entire process. The device/solution always calculate a fingerprint or signature on a variable block size and sees if there is a match. After a block is
processed, it advances by taking another block size and take another blocks and the process repeats.

Figure 4. How the variable block method chunks data.

2.2. Content Aware: Content aware technologies (also called byte level deduplication or delta-differencing de duplication) work in a fundamentally different way. Key element of the content-aware approach is that it uses a higher level of abstraction when analysing the data. Content-aware de-duplication looks at the data as objects. Unlike hash based dedupe, which try to find redundancies in block level, content-aware looks it as objects, comparing them to other objects. (e.g., Word document to Word document or Oracle database to Oracle database.) In this, the deduplication engine sees the actual objects (files, database objects, application objects etc.) and divides data into larger segments (usually 8mb to 100mb in size). Then, typically by using knowledge of the content of the data (known as being “content-aware”), this technique finds segments that are similar and stores only the changed bytes between the objects. That is a BYTE level comparison is performed.

III. PROBLEM STATEMENT

3.1 System and security model

We propose a scheme to deduplicate encrypted data at CSP by applying SHA t issue keys to different authorized data holders based on data ownership challenge. The system contains three types of entities: 1) CSP that offers storage services and cannot be fully trusted. Since it is curious about the contents of stored data. 2) data holder that uploads and saves its data at CSP. The data holders that produces or create the file is regarded as data owner. 3) An authorized party AP that does not collude with CSP and is fully trusted by data holders to verify data ownership and handle dead ringer avoidance.

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(puk,prk)</td>
<td>The public key and private key</td>
</tr>
<tr>
<td>DEKEY</td>
<td>The symmetric key of user</td>
</tr>
<tr>
<td>H()</td>
<td>The hash function</td>
</tr>
<tr>
<td>CT</td>
<td>The cipher text</td>
</tr>
<tr>
<td>M</td>
<td>The user data</td>
</tr>
<tr>
<td>KG</td>
<td>The key generation algorithm</td>
</tr>
<tr>
<td>RG</td>
<td>The re-encryption key generation algorithm</td>
</tr>
<tr>
<td>CK</td>
<td>The cipher key</td>
</tr>
<tr>
<td>K</td>
<td>The secret key</td>
</tr>
</tbody>
</table>

Table 1. System Notation

4.1 Preliminaries

1) Triple DES

Triple Data Encryption Algorithm
a. block cipher with symmetric secret key.
b. block length=64 bits.
c. key length =56, 112 or 168 bits.

Figure 5. Different keys are used to encrypt and decrypt message

3DES was created because DES algorithm, invented in the early 1970s using 56-bit key. The effective security 3DES provides 112 bits due to man in the middle attacks. Triple data encryption standard is a type of computerized cryptography where block cipher algorithm is applied three times to each data blocks. The key size is increased in triple DES to ensure additional security through encryption capabilities. Each blocks contain 64 bits of data. Three keys are referred to as bundle keys with 56 bits per key. There are three keying options in data encryption standards.

a. All keys being independent.
b. Key 1 and key 2 being independent keys.
c. All three keys being identical

(i) Algorithm

Run DES three times:

**ECB Mode:**
If K2=K3, this is DES
Backward compatibility
Known not to be just DES with K4
Hash 112 bits of security, not three 56=168

Triple DES algorithm uses three iterations of common DES cipher. It receives a secret 168-bit key, which is divided into three 56 bit keys.

- Encryption using the first secret key
- Decryption using the second secret key
- Encryption using the third secret key
Using decryption in the second step during encryption provides backward compatibility with common DES algorithm. In these case, first and second secret keys or second and third secret keys are the same whichever key.

\[
C = E_3(D_2(E_1(M))) = E_3(M) \\
C = E_3(D_3(E_1(M))) = E_1(M)
\]

It is possible to use 3 DES cipher with the secret 112-bit key. In this case, first and third secret keys are the same.

\[
C = E_1(D_2(E_1(M)))
\]

4.2 Secure dead ringer avoidance system:

Suppose that the client U1 is the first user who uploads the image Ii to the cloud server. Noted that Ii has to be encrypted firstly before uploading for privacy-preserving, the client U1 applies the key generation algorithm KeyGenCE(Ii) to obtain Ki and encrypts Ii to output the cipher image Ci by the convergent encryption algorithm EncryptCE(Ki, Ii). Then the clients upload Ci to the storage server S1. Meanwhile, the fingerprint of the cipher image H(Ci) is calculated and transferred to both S1 and S2 for deduplication. Deduplication operation. As described above, the deduplication job should be completed by the cloud server before the client uploads his image. Thus, before the client U2 uploads the image file to the storage server S1, he at first sends the fingerprint of encrypted image H(C) to the server side for checking whether the identical copy has already been stored. To compute the fingerprint of encrypted image, the client first generates the key K and encrypts the image by EncryptCE(K, I), then calculates the fingerprint H(C) and sends it to S1 and S2 for checking whether there exists the same fingerprint in H(Ci) with H(C) respectively. Noted that both the encrypted data copy and its fingerprint are stored on the S1 server side. Verifiability of deduplication in our scheme as described above, if U1 is the first client to upload a cipher image Ci, he must also transfer the fingerprint H(Ci) to S1 and S2. Thus, once the server S1 or S2 have found the same fingerprint in H(Ci) with H(C), it implies that the same image has already been uploaded to the server by another client U1 before. So, the corresponding server will reply to the client U2 about this result.

4.3 Hash function

Hash functions are extremely useful and appear in almost all information security applications. A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length. Values returned by a hash function are called message digest or simply hash values.
The typical features of hash functions are –

- **Fixed Length Output (Hash Value)**
  Hash function converts data of arbitrary length to a fixed length. This process is often referred to as hashing the data. In general, the hash is much smaller than the input data, hence hash functions are sometimes called compression functions.
  Since a hash is a smaller representation of a larger data, it is also referred to as a digest. Hash function with n bit output is referred to as an n-bit hash function. Popular hash functions generate values between 160 and 512 bits.

- **Efficiency of Operation**
  Generally, for any hash function h with input x, computation of h(x) is a fast operation. Computationally hash functions are much faster than a symmetric encryption.

V. CONCLUSION:

Managing encrypted data with deduplication is important and significant in practice for achieving successful cloud storage service, especially for big data storage. In this paper, we proposed a practical scheme to manage the encrypted big data in cloud with deduplication based on ownership challenge and Triple DES. Our schemes can flexibly support data update and sharing with deduplication even when the data holders are offline. Encrypted data can be securely accessed because only authorized data holders can obtain the symmetric keys used for data decryption. Extensive performance analysis and test showed that our scheme is secure and efficient under the described security model and very suitable for big data deduplication. The results of our computer simulations further showed the practicability of our scheme. Future work includes optimizing our design and implementation for practical deployment and studying verifiable computation to ensure that CSP behaves as expected in deduplication management.

VI. REFERENCES


