CLOUD RECOVERY THROUGH INDEXING

Balkrushna B. Jagadale[1], TruptiA.Dhumal[2] & Rangnath D. Kulkarni[3]

[1].Assistant Professor,ComputerScience& Engineering Dept.

Sveri Collage of Engineering, Pandharpur.

[2]Assistant Professor, ComputerScience & Engineering Dept.

Sveri Collage of Engineering, Pandharpur.

[3]Assistant Professor, ComputerScience & Engineering Dept.

Sveri Collage of Engineering, Pandharpur.

Abstract: In cloud environment, large amount of data can be stored. To maintain the efficiency of those data needs recovery services. In cloud computing large amount of private data stored on the main cloud. Therefore the necessity of the recovery services growing day- by -day and it requires a development of an efficient and effective data recovery service technique. The purpose of the recovery technique is to help user to collect information from any backup server when the server fails to provide the data to the user. There are lots of recoveries mechanisms are used to recover the data in the cloud such as HSDRT, ERGOT, LINUX BOX, PCS, COLD and COLD/HOT backup strategy. But there are some limitations in those techniques such as implementation complexity, security issues and retrieval time is high. That is the main issues in existing systems; to overcome from these issues we can use Master File Table (MFT) data storage with its index through recovery.

Keywords: Cloud Computing, Disaster Management, Index Based Recovery, Master File Table.

1. INTRODUCTION

In the public cloud model, a third-party provider delivers the cloud service over the Internet. Public cloud services are sold on-demand, typically by the minute or the hour. Customers only pay for the CPU cycles, storage or bandwidth they consume. Leading public cloud providers include Amazon Web Services (AWS), Microsoft Azure, IBM/Soft Layer and Google Compute Engine. Although cloud computing has changed over time, it has always been divided into three broad service categories: infrastructure as a service (IaaS), platform as a service (PaaS) and software as service (SaaS). In this paper we are going to implement a recovery technique that will recover data through a crashed Cloud using the previous memory index of that particular data or file which you want to recover. We are going to maintain the MFT Record in back-up system of a cloud.

2. Existing Work

The recent back-up and recovery techniques that have been developed in cloud computing domain such are HSDRT, (PCS) Parity Cloud Service, (ERGOT) Efficient Routing Grounded on Taxonomy, Linux Box, Cold/Hot backup strategy etc. Detail review shows that these techniques are able to provide best performances under all uncontrolled circumstances such as cost, security, low implementation complexity, redundancy and recovery in short span of time.

3. Proposed Work

Proposed work will be implemented for the data recovery from the cloud if that cloud is crashed. We are going to study a recovery technique that will recover data through a crashed cloud using the previous memory index of that particular data or file which you want to recover.

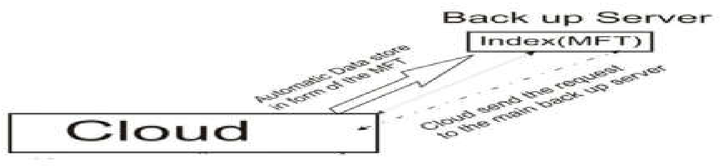

Back up Server
Index(MFT)
Automatic Data store in form of the MFT
Cloud send the request to the main back up server
Cloud

Figure 1. Architecture

The cloud backup server stores the data in the form of the **MFT (Master File Table)** format as a backup. Means it store the only Index record of a main clouds; it don't store the whole data or replicas of a cloud. Backup server and the clients are does not interact with each other. The main advantages of this backup sever module is large amount of data in index format. When main cloud/server will lost or corrupt then main server will send request to the backup server. Backup server then recover that whole data by using indexing and after completing this recovery process send it to the server/cloud data by using seed block algorithm.

4. Algorithms Used

For this study we are going to use 3 different algorithms as,

4.1. Seed Block Algorithm (SBA).
4.2. Data Encryption Standard (DES).
4.3. Huffman Algorithm.

4.1. Seed Block Algorithm (SBA).

This algorithm focuses on simplicity of the back-up and recovery process. It basically uses the concept of Exclusive– OR (XOR) operation of the computing world. For ex: - Suppose there are two data files: A and B. When we XOR A and B it produced X i.e. X =⨁. If suppose A data file get destroyed and we want our A data file back then we are able to get A data file back, then it is very easy to get back it with the help of B and X data file .i.e. A = ⨁. Similarly, the Seed Block Algorithm works to provide the simple Back-up and recovery process. Its architecture is shown in Fig-2 consists of the Main Cloud and its clients and the Remote Server. Here, first we set a random number in the cloud and unique client id for every client. Second, whenever the client id is being register in the main cloud; then client id and random number is getting EXORed (⨁) with each other to generate seed block for the particular client. The generated seed block corresponds to each client is stored at remote server. Whenever client creates the file in cloud first time, it is stored at the main cloud. When it is stored in main server, the main file of client is being EXO Red with the Seed Block of the particular client. And that EXORed file is stored at the remote server in the form of file' (pronounced as File dash). If either unfortunately file in main cloud crashed / damaged or file is been deleted mistakenly, then the user will get the original file by EXORing file' with the seed block of the corresponding client to produce the original file and return the resulted file i.e. original file back to the requested client. The main purpose of seed block algorithm is it takes the minimum time for remotely access the data of recovery process.

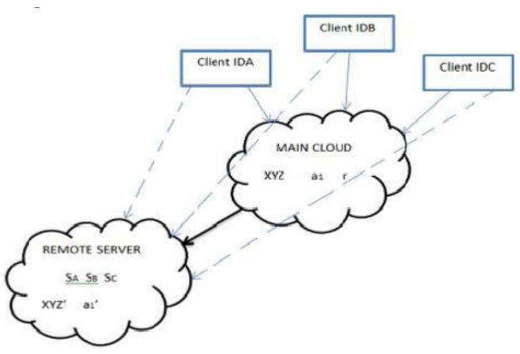
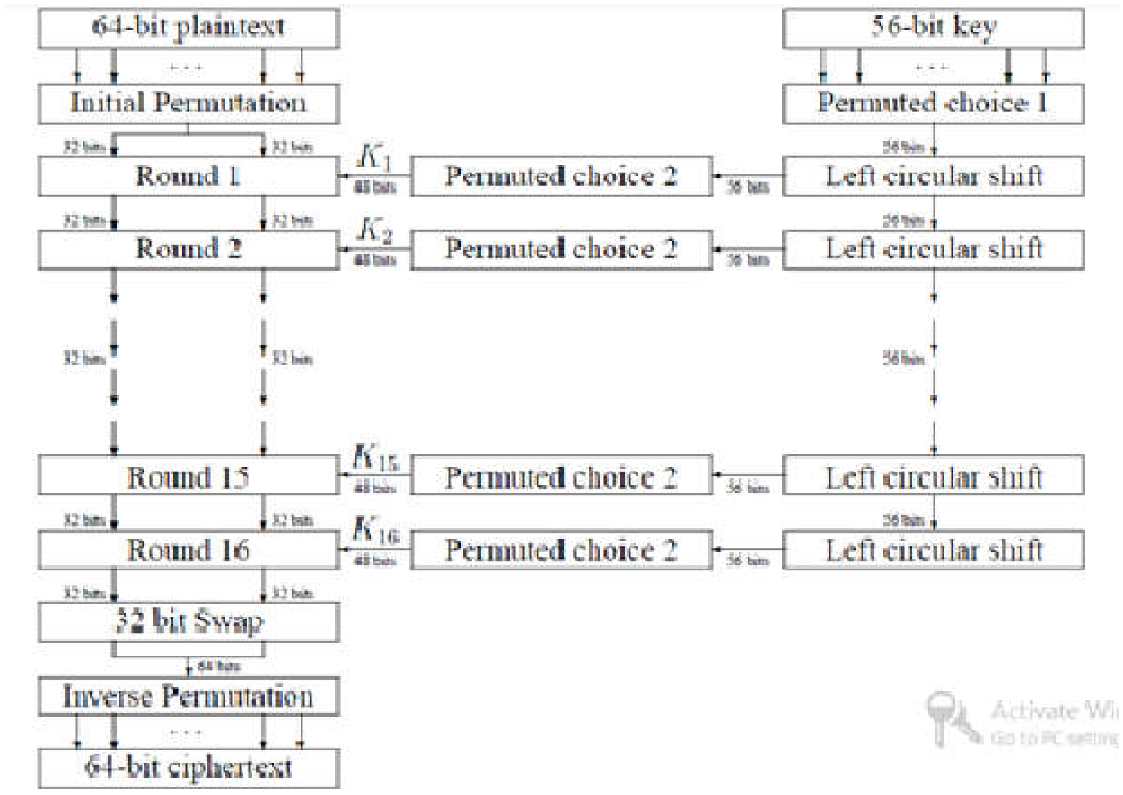


Figure 1. SBA Architecture

SBA Algorithm Is As Follows:

Initialization: Main Cloud: Mc Remote Server: Rs Clients of Main Cloud: Ci; Files: $a1$ and $a1'$ Seed Block: Si; Random Number: r; Client's ID: $Client\_Idi$
Input: $a1$ created by Ci; r is generated at Mc;
Output: Recovered File $a1$ after deletion atMc;
Given: Authenticated Clients could allow uploading, downloading and do modification on its own the files only.
Step 1: Generate a random number. Int r= rand ( );
Step 2: Create a Seed Block Si for each Ci and store Si at Rs, Si= r$\oplus Client_Idi$ (Repeat step 2 for all clients)
Step 3: If Ci/ Admin Creates /modifies a $a1$and Stores at , then $a1'$ create $a1'$=$a1 \oplus Si$
Step 4: Store $a1'$ at Rs Step 5: If Server crashes $a1$ deleted from Mc, then we do EXOR to retrieve the original $a1$ as: $a1$ =$a1' \oplus Si$
Step 6: Return $a1$ to Ci
Step 7: END.

4.2. Data Encryption Standard.
Data Encryption Standards are used for encrypting and decrypting of data while uploading to cloud and downloading from cloud.



DES Algorithm as follows:

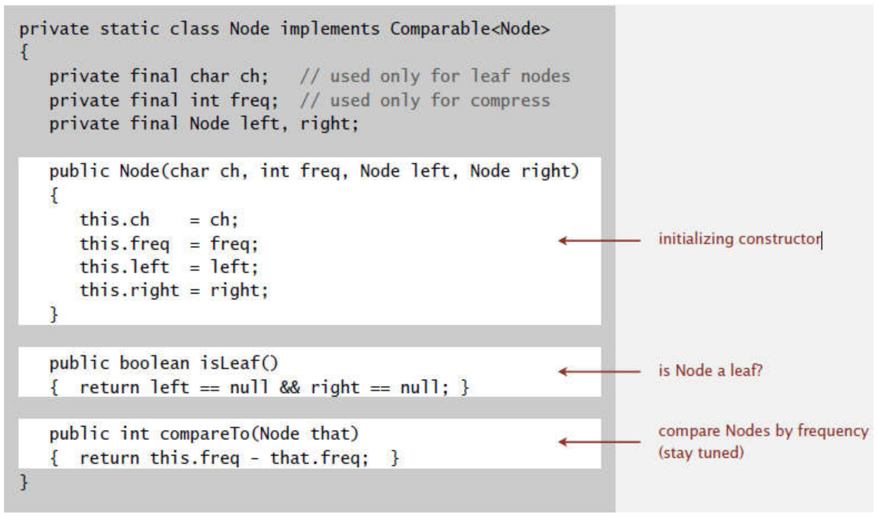
1. Initial permutation (**IP** - defined in table 2.1) rearranging the bits to form the "permuted input".
2. Followed by 16 iterations of the same function (substitution and permutation).The output of the last iteration consists of 64 bits which is a function of the plaintext and key. The left and right halves are swapped to produce the pre output.

3. Finally, the pre output is passed through a permutation (**IP**−1 - defined in table 2.1) which is simply the inverse of the initial permutation (**IP**). The output of **IP**−1 is the 64-bit cipher text Table.

4.3. Huffman Algorithm.
Compression:
- Read message.
- Built best prefix-free code for message. How?
- Write prefix-free code (as a trie) to file.
- Compress message using prefix-free code.

Expansion:
- Read prefix-free code (as a trie) from file.
- Read compressed message and expand using trie.

Huffman trie node data type:-



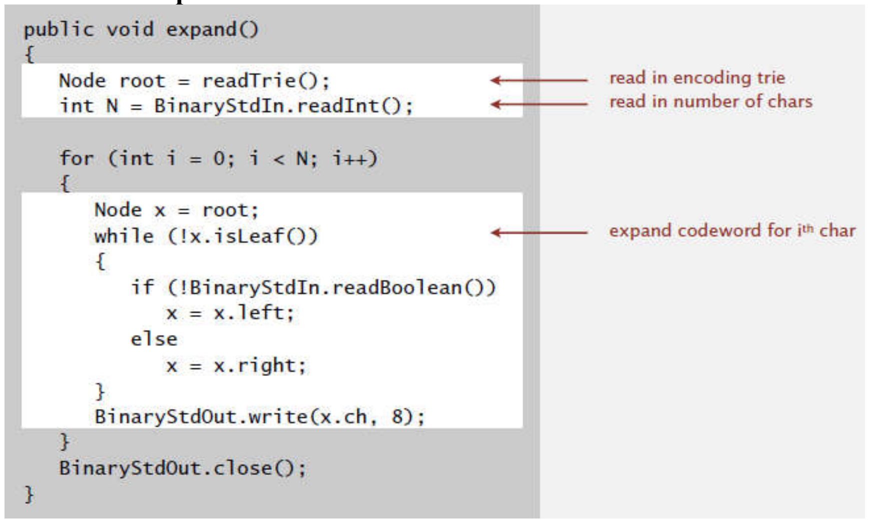
```
private static class Node implements Comparable<Node>
{
   private final char ch;   // used only for leaf nodes
   private final int freq;  // used only for compress
   private final Node left, right;

   public Node(char ch, int freq, Node left, Node right)
   {
      this.ch    = ch;
      this.freq  = freq;
      this.left  = left;
      this.right = right;
   }

   public boolean isLeaf()
   {  return left == null && right == null; }

   public int compareTo(Node that)
   {  return this.freq - that.freq;  }
}
```

Prefix-free codes: expansion:-

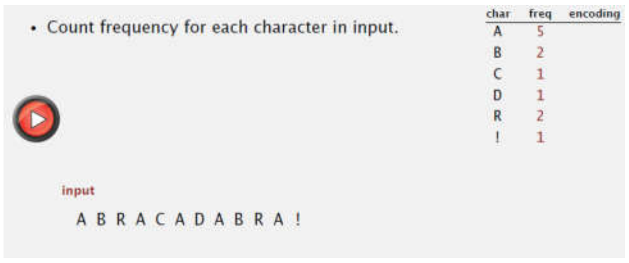


```
public void expand()
{
   Node root = readTrie();
   int N = BinaryStdIn.readInt();

   for (int i = 0; i < N; i++)
   {
      Node x = root;
      while (!x.isLeaf())
      {
         if (!BinaryStdIn.readBoolean())
            x = x.left;
         else
            x = x.right;
      }
      BinaryStdOut.write(x.ch, 8);
   }
   BinaryStdOut.close();
}
```

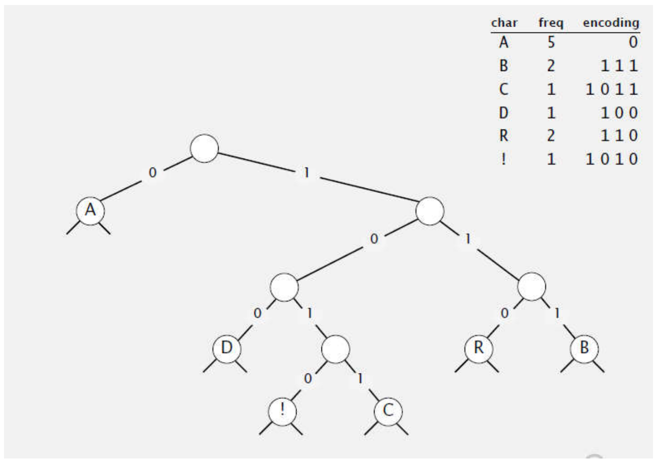
Running time. Linear in input size *N*.

Huffman algorithm demo:-



• Count frequency for each character in input.

char	freq	encoding
A	5	
B	2	
C	1	
D	1	
R	2	
!	1	



char	freq	encoding
A	5	0
B	2	111
C	1	1011
D	1	100
R	2	110
!	1	1010

Huffman algorithm:

1. Count frequency freq[i] for each char i in input.
2. Start with one node corresponding to each char i (with weight freq[i]).
3. Repeat until single trie formed:
   – select two tries with min weight freq[i] and freq[j]
   – merge into single trie with weight freq[i] + freq[j]

5. CONCLUSION

The study is for computerizing the working in Cloud Computing. We can successfully recover data from corrupted cloud through indexing in clod computing. If it will be implemented for real time, even if hacker hacks the data from the server or someone stoles the data or damage the cloud infrastructure then there will not be a data loss for the server.

REFERENCES

1._Disaster Recovery System Using Seed Block Algorithm in Cloud Computing Environment (IEEE Paper November 2012. 1. S.Deepa –Dept. of CSE Annamalari University, 2. Dr. G. Ramachandram.-Dept of CSE Tamil Nadu, India)._

2. _Data Back-Up and Recovery Techniques for Cloud Server Using Seed Block Algorithm. (IEEE paper February 2015. 1. R. V. Gandhi, (Dept. CSE,Jigjija university, Jigjija,Ethiopia). 2.MSeshaiah,(Dept. CSE,Visvesvaraya Technological University,Belgum,India) 3. A.Srinivas,(Dept. CSE,HITS,R.R.Dist,Telangana,Indai) 4. C. ReddiNeelima (Dept. CSE, MTIEAT, palamaner, Chittor Dist., A.P, India)._