ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.5

Managing Cloud Storage using a Cost Efficient Cuckoo Hashing Scheme

Priyanka Pujari[1], Chaitra D Shivanagoudar[2], Mohini M Khokane[3], Kavita B Kudachi[4], Shilpa M Gonsalves[5]
Assistant Professor[1], BE student[2, 3, 4, 5]
Department of Computer science and Engineering
Angadi Institute of Technology and Management, Belagavi, India

Abstract:
Cloud computing servers need to process and analyze large amounts of system resources. This usually requires many query operations like insertion, deletion and searching. Due to simplicity and ease of use, cuckoo hashing schemes have been widely used in real world cloud-related applications. Cuckoo hashing suffers from endless loops and high insertion latency, due to the potential hash collisions. To address these problems, a cost efficient cuckoo hashing scheme, called MinCounter has been proposed. MinCounter reduces the occurrences of endless loops in the data insertion by selecting busy kicking-out routes. MinCounter selects "COLD", rather than random buckets to handle hash collisions. MinCounter offers efficient insertion and provides high performance to cloud servers and enhances the experience for cloud users. MinCounter can efficiently improve the utilization of cuckoo hash tables and decrease the rehash probability to increase the cloud computing system performance. Extensive experimental results demonstrate the efficiency of MinCounter.

Keywords: Cuckoo hashing, Cloud computing, Hash Collisions, Latency, MinCounter.

I. INTRODUCTION

The name cuckoo hashing is derived from the bird "Cuckoo" which means the cuckoo chick pushes the other eggs or young out of the nest when it hatches, while inserting a new key into a cuckoo hashing table may push an older key to a different location in the table. It is an open addressing form in which the non empty cell of hash table has key or key value pair , cuckoo hashing solves the problem of collision by using two hash functions instead of using only one, the hash table is split into two smaller tables , which are of equal size and each function provides an index into one of these two tables and it is also possible to provide in indexes into a single table ,inserting a key can be done only when the cell is empty, if the both the cell is full than the other key as to moved to the second location to make place for the new key, the new key can be placed by kicking out the old key and thus by replacing the new key, this process continues until an empty position is found and suppose the insertion fails than infinite loop may occur.

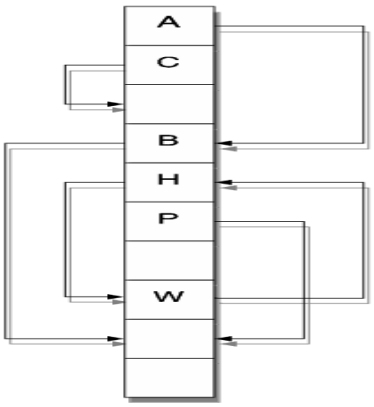

Figure.1. Insertion process in cuckoo hashing table

The new item can be inserted in location A by kicking out A to other location that is B and moving B to some other location which is vacant. H is a part of cycle, so H will not succeed in insertion of new item and new item would be kicked out again as shown in the above figure1.

DETAILS OF CUCKOO HASHING

Cuckoo hashing contains bucket that is used for inserting items and reduces hash collision, cuckoo hashing scheme improves space utilization

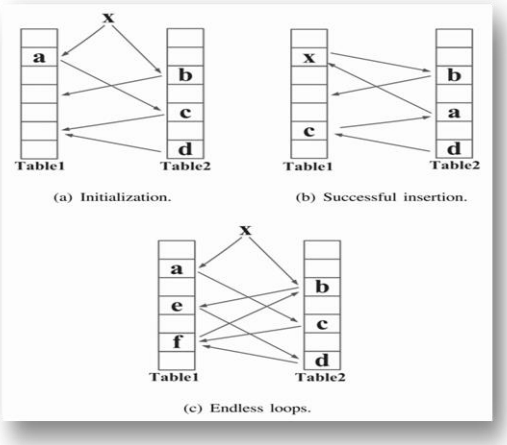

Figure.2. Insertion process in cuckoo hashing

- Its important to check whether there is any empty bucket or no, if not than the original item should be kicked out thus by replacing the new one
- The kick out item is inserted into table 2 ,this process is continued until all items find bucket
- In figure 2(b) shows that the item is inserted successful into the table 1by moving items from one table to other
- In the figure 2(c) shows the occurances of endless loop when the item fails to find the bucket. The cuckoo hashing fails to fully avoid the hash collisions. Thus a min-counter scheme is used.

MINCOUNTER

Min-counter is used because the cuckoo hashing scheme suffers from endless loop and high insertion latency, and it has high risk of reconstruction of hash table. Min-counter selects cold bucket that is infrequently used to handle hash collision rather than random buckets.

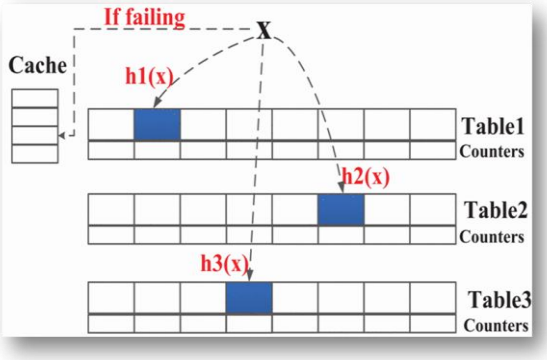

Figure .3. MinCounter

As shown in figure 3 in each bucket is allocated by table the counter is used to record the kicking out times of buckets, when there is the occurance of hash collision in the bucket a the corresponding counter is increased by 1. If the item is inserted without the availability of empty bucket, thus choose the bucket with minimum counter to execute the replacement.

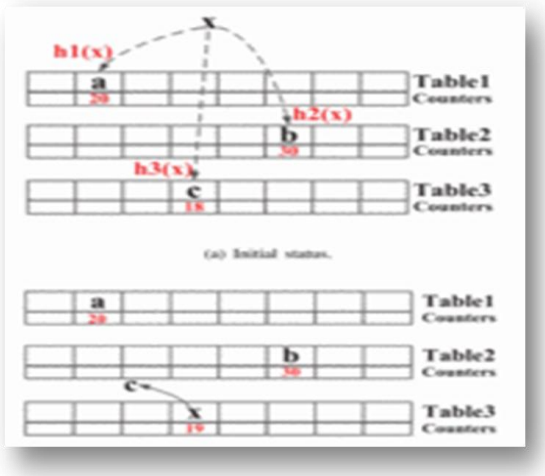

Figure .4. Insertion Details

Figure 4 shows the insertion details. When the item is inserted into the hash table it is necessary to check empty bucket of h1(x) ,h2(x),h3(x), each candidate bucket of x is occupied by a,b,c thus by comparing the counters of candidate buckets choose the minimum one (i.e..,18) and replace item c with x. next the counter of the bucket of h(x) increases by 1 upto 19, the kick-out item becomes one needed to inserted, and insertion procedure goes on, until an empty slot is found in hash tables.

II. EXISTING SYSTEM

The word "cloud" is used as a metaphor for the internet, so the phrase cloud computing means a type of Internet-based computing, where different services- including servers, storage and applications are delivered to an organization's computers and devices through the Internet. Although cloud computing systems consume a large amount of system resources, it is still challenging to obtain accurate results for query results in a real-time manner. In order to improve entire performance and storage efficiency, existing schemes have been proposed, such as **hierarchical Bloom Filter** index to speed up the searching process, **continuously monitoring query execution** to optimize the could scale query ,**Query optimization for parallel data processing, multi-keyword search over encrypted cloud data, similarity search in file systems.**

DISADVANTAGES OF EXISTING SYSTEM

- **Space inefficiency**

The existing system consumes for space due to which we suffer from the space inefficiency.

- **High complexity hierarchical addressing**

The existing system contains many hierarchical addressing which is complex in nature and difficult to understand.

- **High insertion latency**

The cuckoo hashing schemes based on random-walk approach migrate items randomly. Mincounter scheme for cloud storage systems to migrate the actual hash collisions and high-latency in the insertion process.

- **Intensive data migration**

When new data items are inserted into storage servers via cuckoo hashing, a kicking-out operation may incur intensive data migration among servers. The kicking-out operation needs to migrate a selected item to its other candidates and kick out another existing item until an empty slot is found. Frequent kicking-out operations cause intensive data migration among multiple buckets of hash tables. Conventional cuckoo hashing based schemes heavily depend on the timeout status to identify an insertion failure.

III. PROPOSED SYSTEM

Cuckoo hashing addresses hashing collision via simple "kicking-out" operations, which moves items among hash tables during insertions, rather than searching the linked lists. In order to prevent formations of endless loop in hash tables a cost efficient cuckoo hashing scheme called Mincounter is used. The idea behind Mincounter is to alleviate the occurrence of endless loops in the data insertion. Mincounter selects "cold" buckets to handle hash collisions rather than random buckets. Mincounter allows items to be inserted into hash tables to improve the storage space efficiency.

ADVANTAGES OF PROPOSED SYSTEM

- **Reducing hash collisions**

Mincounter alleviate the occurrence of endless loops for large-scale cloud computing systems. Mincounter takes advantage of "cold" buckets to alleviate hash collisions and decrease insertion latency.

- **Improving Space Efficiency and Decreasing Insertion Latency**

Mincounter demonstrates salient performance superiority in terms of insertion latency. It achieves load balance by kicking items out to "cold" positions when hash collisions occur. We can mitigate data collisions and reduce total times of kicking-out operations. Mincounter hence improves space efficiency and decreases insertion latency.

IV . SYSTEM DESIGN

In the proposed system the data owner tries to upload the a file in the web server. By using RSA algorithm two keys are generated one is public key and other is a master secret key. This secret key is used for encryption purpose. During index handling phase the unwanted keywords in the file and important keywords are separated which are already maintained in the database. Hashing is done to the important keywords by using MD5 algorithm and stored in database

using min counter technique. Now the encrypted file will be successfully uploaded in the cloud. When a user tries to download the file he uses a single keyword to search for the desired file. Weight calculating is done for the files which contain the desired keyword and ranking is given based on the number of times the keyword appearing in that particular files. Now the data user can the private key and download the file which he wants in his system.

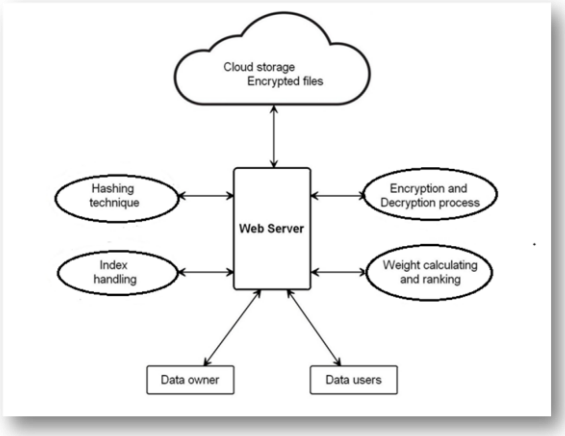

Figure .5. The various stages a file or data undergoes

V. SYSTEM SPECIFICATION

HARDWARE REQUIREMENTS

System	**:**	Pentium IV 24 GHz
Hard disk	**:**	500 GB
Ram	**:**	4 GB

Any desktop/laptop system with above configuration or higher level

SOFTWARE REQUIREMENTS:

Operating System	**:**	Windows XP/7/8
Coding Language	**:**	Java (jdk 1.7)
Web Technology	**:**	Servlet, JSP
Web Server	**:**	TomCAT 7.0
IDE	**:**	Eclipse Galileo
Database	**:**	MYSQL 5.0
UGI for DB	**:**	SQLyog
JDBC Connection	:	Type 4

VI . SNAPSHOTS

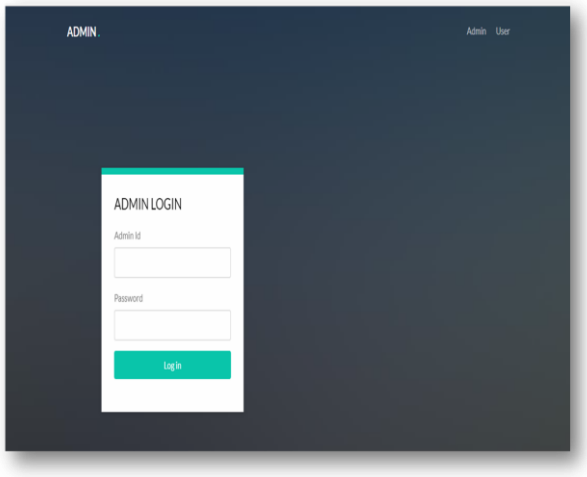

Figure .1. Admin login page

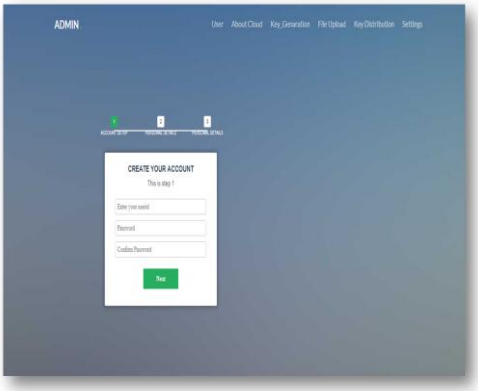

Figure.2. Creating user account

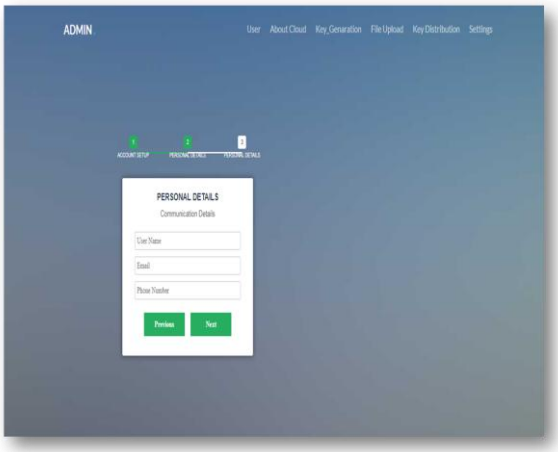

Figure .3. User Personal Details

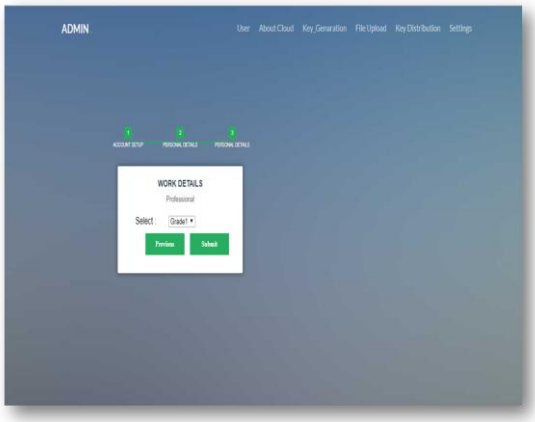

Figure .4. Users Work Details

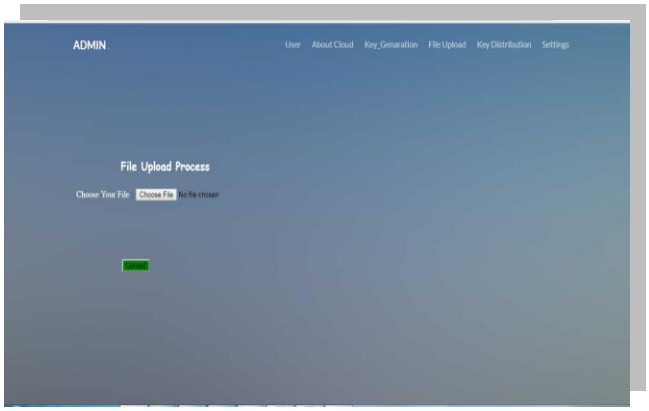

Figure .5. file upload process

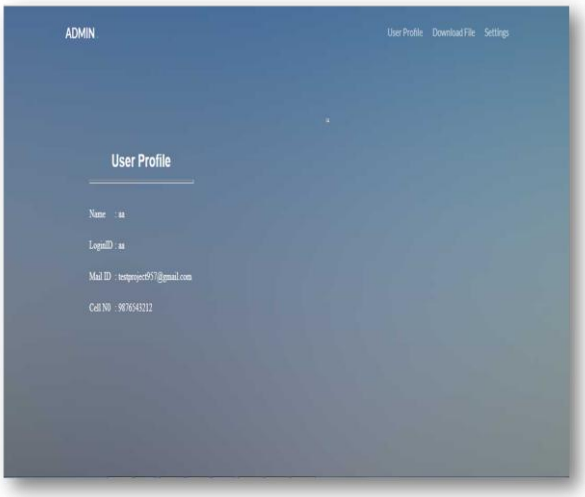

Figure .6. Showing the User Profile

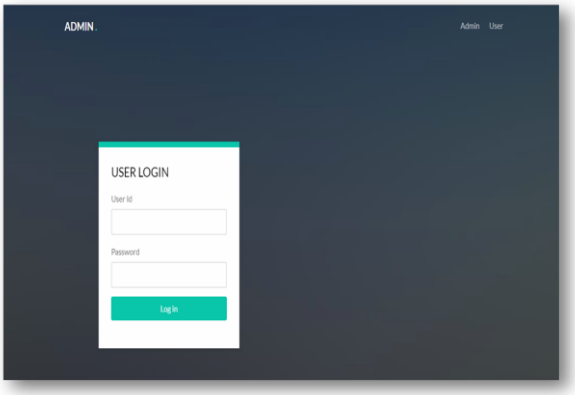

Figure .7. User Login Page

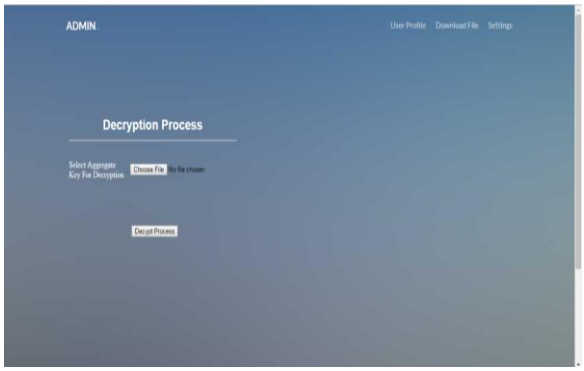

Figure. 8. File Decryption process

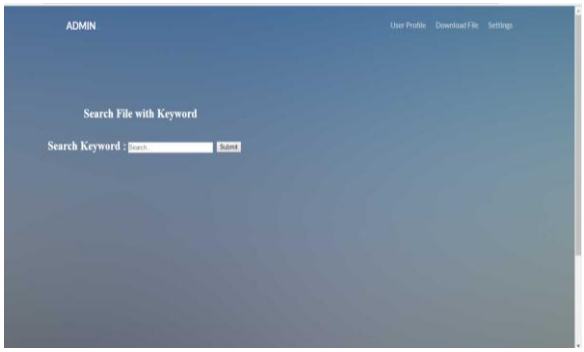

Figure .9. Searching file using keyword

VII. CONCLUSION AND FUTURE SCOPE

To reduce the occurrence of endless loops and high insertion latency, a cost-efficient cuckoo hashing scheme have been proposed called MinCounter. MinCounter reduces the occurrences of endless loops in the data insertion by selecting busy kicking-out routes. MinCounter selects "COLD", rather than random buckets to handle hash collisions. MinCounter takes the advantages in terms of the utilization ratio of hash tables, the total kicking out times and its optimizes the performance for cloud servers. For future scope is to mitigate the actual hash collisions and high-latency in the insertion process.

VIII. ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of this project would be incomplete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. We consider our self privileged to express gratitude and respect towards all those who guided us through the completion of this project. We convey thanks to our project guide **Prof. Priyanka Pujari,** Department of Computer Science and Engineering, AITM, Belagavi for providing encouragement, constant support and guidance which was of a great help to complete this project successfully. We are grateful to **Prof. Sagar Birje**, Head, Department of Computer Science and Engineering for giving us the support and encouragement that was necessary for the completion of this project. We would also like to express our gratitude to **Dr. Sanjay Pujari**, Principal and Director, Angadi Institute of Technology and Management for his support and encouragement.

IX. REFERENCES

[1]. Mohamed Hamdi "Security of Cloud Computing, Storage, and Network", School of Communication Engineering, Technopark El Ghazala, 2083 Tunisia, IEEE Paper 2012.

[2]. Nicolas Bruno, Sapna Jain IIT Bombay JingrenZhou "Continuous Cloud Scale Query Optimization and Processing" Microsoft Corp, IEEE Paper 2013.

[3]. Qiuyu Li, Yu Hua, Wenbo He, Dan Feng, Zhenhua Nie, Yuanyuan Sun,Wuhan "Necklace: An Efficient Cuckoo Hashing Scheme for Cloud Storage Services", National Lab for Optoelectronics, School of Computer Huazhong University of Science and Technology, Wuhan, China School of Computer Science McGill University, Montreal, Quebec, Canada, IEEE Paper 2014.

[4]. Yu Hua, Hong Jiang, Dan Feng "FAST: Near Real-time Searchable Data Analytics for the Cloud", Wuhan National Lab for Optoelectronics, School of Computer Depart. Computer Science and Engineering WNLO, School of Computer Huazhong, IEEE Paper 2014.

[5]. Yuanyuan Sun, Yu Hua, Dan Feng, Ling Yang ,Pengfei Zuo, Shunde Cao, Wuhan "MinCounter: An Efficient Cuckoo Hashing Scheme for Cloud Storage Systems" National Lab for Optoelectronics, School of Computer Huazhong University of Science and Technology, Wuhan, chaina IEEE paper 2015.

[6]. Mohit Tiwari, Rakshit Mahajan,Shradda Ahuja, Sahil Rawat &Vishrut Mittal "Fuzzy Keyword Search over Encrypted in cloud computing.", Department of Computer Science Engineering, GGSIPU Rohtak Road, East Paschim Vihar Delhi, India, IEEE Paper 2016.

[7]. Shadab Ahmad "An Efficient Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Data in Cloud",

School of Computer and Information Science University Of Hyderabad Hyderabad, IEEE Paper 2016.

[8]. Choong Hee Cho*, JungBok Lee, and Jeong-dong Ryoo "A Collision-Mitigation Hashing Scheme Utilizing Empty Slots of Cuckoo Hash Table", Information and Communication Network Technology, Korea University of Science and Technology (UST), Daejeon 34113,Korea Kakao Corp, Sungnam 13494, Korea, IEEE Paper 2017.