International Journal of Scientific Research & Engineering Trends
Volume 4, Issue 3, May-June-2018, ISSN (Online): 2395-566X

A collision-Mitigation cuckoo hashing scheme for large scale storage systems

Asst.Prof. Mrs.Kavyashree .C
Dept. of CSE, MVJCE computer Science and engineering
Bangalore, India

M.Tech.Schalor Bhagyalakshmi H N
Dept. of CSE, MVJCE computer Science and engineering
Bangalore, India
bhagyalakshmi.hn39@gmail.com

***Abstract*- Cloud computing systems consume an outsized quantity of system resources; it's still difficult to get correct results for question requests in a real-time manner. Due to space inefficiency and high complexity hierarchical addressing schemes fail to meet the wants of real time queries. So as to support real-time queries, hashing based information structures are widely utilized in constructing the index due to constant-scale addressing complexity and fast query response. Unfortunately hashing based information structures cause low space utilization, as well high latency risk of handling hashing collisions. Traditional techniques used in hash tables to deal with hash collisions include open addressing, chaining and coalesced hashing. Cuckoo hashing addresses hashing collisions via simple "kick-out" operations, which moves items among hash tables during insertion, rather than searching the linked list (i.e. hierarchical addressing).**

***Keyword*- hash collisions, cloud computing, system.**

I.INTRODUCTION

Cloud information proprietors like to outsource archives in a scrambled shape with the end goal of access control framework. In this way it is fundamental to create proficient and reliable cipher text search techniques. One test is that the connection between reports will be typically covered during the time spent encryption, which will prompt critical pursuit exactness execution debasement.

Likewise the volume of information in server farms has encountered a sensational development. This will make it even more challenging to design cipher text search schemes that can provide efficient and reliable online information retrieval on large volume of encrypted data.

Despite the fact that distributed computing frameworks devour a lot of framework assets, it is still difficult to acquire precise outcomes for question asks for in a constant way. Keeping in mind the end goal to enhance whole framework execution and capacity productivity, existing plans have been proposed.

For example, various leveled Bloom channel file to accelerate the looking procedure , consistently checking question execution to enhance the cloud-scale inquiry, inquiry streamlining for parallel information preparing , estimated enrollment question over cloud information , multi-watchword look over scrambled cloud information, closeness seek in document frameworks , limiting recovery idleness for content cloud data and recovery for positioned inquiries over cloud information.

Because of space wastefulness and high-complexity hierarchical addressing, these plans neglect to address the issues of constant inquiries. So as to help ongoing inquiries, hashing-based information structures have been generally utilized as a part of building the list because of steady scale tending to many-sided quality.

Tragically, hashing-based information structures cause low space use, and additionally high-latency danger of dealing with hashing crashes. Customary strategies utilized as a part of hash tables to manage hashing impacts incorporate open tending to hashing.

Dissimilar to ordinary hash tables, cuckoo hashing addresses hashing impacts by means of straightforward "kicking-out" activities (i.e., level tending to), which moves things among hash tables during insertions, rather than searching the linked lists (i.e., hierarchical addressing).

II.SYSTEM DESIGN

1. System architecture

System architecture helps in understanding the complete work done easily in the project. There is likewise a requirement for the remapping rationale to

divert all frontend LBAs (logical block addressing) to their genuine positions on the circles, since obstructs with various LBAs may have similar information content.

In our record table plan, we utilize the MD5 hash, for its impact safe properties, to process the unique finger impression. Altogether, there are 16 bytes in every passage with 128 bits of yield for this hash work.

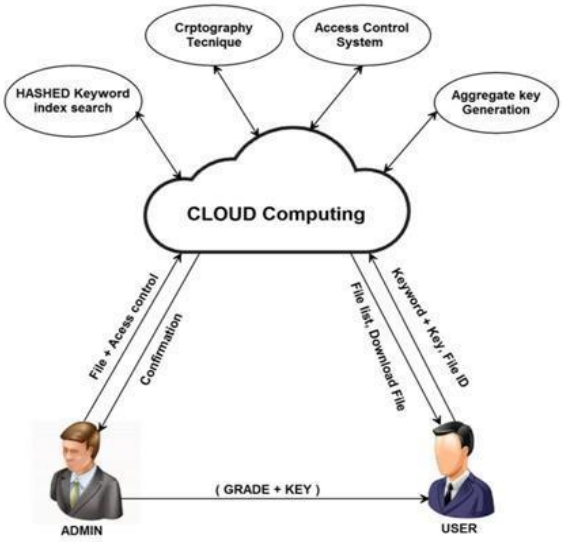

Fig.1 admin logs into the cloud by providing username and password

As shown in the above diagram the admin logs into the cloud by providing username and password admin as the file access permissions. If the login details are proper cloud provides the confirmation, admin creates the user account with the necessary details and provides the password to user.

A cryptographic technique is used to encrypt and decrypt the file .when admin uploads the file to cloud user decrypt the file by providing the secrete key .the hash code is generated and send to the mail id of the user the user enter the file name and decrypt the file and searches the file in the index by using the keyword.

The proposed architecture helps in uploading the large files in the cloud and ensures the security, this proposed system can be used in large organizations, it also achieves the security where the 3rd parties can't hack the file because a strong security is provided.

2. Hash Collision

A hash work takes a thing of a given sort and produces a whole number has an incentive inside a given range. The information things can be anything: strings, accumulated shade program document even catalogs.

Similar information dependably produces similar hash esteem, and a decent hash work has a tendency to create diverse hash esteems when given distinctive sources of info. Here the multilevel hashing system creates great and diverse hash esteems for various sources of info.

3. System Modules

Data Owner Session

- Login.
- Configuration.
- File Storage Server Configuration.
- Key Setting (RSA).
- Upload a file.
- Remove un-necessary words.
- Find Keywords.
- Encrypt the File using Private Key.
- Upload the encrypted file into the file storage.

4. Keyword Ranking

- View keyword & weight age.
- Calculate the Ranking for keyword.
- Store the keyword Ranking in a Table.
- View keyword Ranking details.
- View User Detail.
- User Request.
- View user Request.
- Accept / Reject user request.
- Send Public Key through email (RSA Algorithm).
- Change Password.

5. Data User Session

- User Registration.
- Login.
- User Profile.
- Request Management.
- Send key request to server.
- View the Request Status.
- Search with Keyword.
- Input Search Keyword.
- Send the request to server.

III. IMPLEMENTATION

1. Multi-keyword Module

This module is utilized to help the client to get the exact outcome in view of the different watchword ideas. The clients can enter the different words inquiry, the server will part that question into a solitary word after hunt that word document in our database.

At last, show the coordinated word list from the database and the client gets the record from that rundown.

2. Algorithm and Methodology

Hashing Technology has been used for Data Integrity process in proposed system.

- Term Frequency Algorithm is used to shortlist the keywords.
- Inverse Document Frequency Algorithm is used to Pick TOP K relevant Files.
- Asymmetric Cryptography Algorithm is used for file content encryption and decryption.
- Diffie-hellmen key exchange algorithm.
- File Transfer Protocol is used for file uploading and downloading.

3. Proposed algorithms

In proposed algorithm involves following processes,

- Cuckoo Hashing Technique.
- Document Encryption.
- Document uploading process.
- Data mining Process.
- Document parse.
- Grouping the documents based on Keyword Weight age.
- Data Creation.
- Keyword Weight age Index Maintenance.
- Search using Multiple Keywords.

4. Efficient Ranked Searchable scheme with user feedback weight age

- File Download.
- File Decryption.
- User feedback on Search result.

5. Term frequency and inverse document frequency

Term recurrence calculation and IDF (converse record recurrence) calculation is utilized to look through the keyword and shortlist the keyword. Tf-idf, short for term frequency– converse archive recurrence, is a numeric measure that is use to score the significance of a word in a report in view of how frequently did it show up in that record and a given gathering of records.

The instinct for this measure is: If a word shows up as often as possible in an archive, at that point it ought to be imperative and we should give that word a high score. Be that as it may, if a word shows up in excessively numerous different records, it's presumably not a novel identifier; along these lines we ought to appoint a lower score to that word. The math recipe for this measure.

- Tf id f(t,d,D)=tf(t,d)×idf(t,D).
- t – terms.
- d- document.
- D –collection of documents.

6.Cuckoo hashing algorithm to insert items

To insert (item x)

1. Hash1(x)
2. a Hash1(x)
3. t Determine V-add(a,b)
4. if t==v+2 at that point
5. Assign a special ID to the new sub diagram
6. Union (a,b)
7. Direct Insert(x,a,b)
8. Return True/*Finish the addition */
9. Else if t==V+1 at that point
10. Union (a,b)
11. Direct insert(x,a,b)
12. Return genuine/* Finish the insertion*/
    ElsE
13. Indirect insert(x,a,b)
    end if

7.Algorithm to insert the items directly into the table

Algorithm2

Direct insert(Item x,Hash a, Hash b**)**

1. /* an and b are two applicant position of thing x */.
2. If B[a] is vacant at that point.
3. B[a].
4. Else.
5. B[b].
6. end if.

8. Asymmetric Cryptography Algorithm

Here in this project a secure RSA algorithm is been used to encrypt and decrypt the files in the cloud.RSA is a secure file transmission algorithm. Secure RSA keeps documents from programmers and help safe transmission of records from one end to other.

1. Pick four extensive prime numbers p, q, r and s haphazardly and freely of each other. All primes ought to be of equal length.
2. Process n = p x q, m= r x s, φ= (p-1) x (q-1) and λ=(r1) x(s-1).
3. Pick a whole number e, 1 < e < φ with the end goal that Gcd (e, φ) =1
4. Process the mystery example d, 1 < d < φ, with the end goal that e x d mod=1.
5. Select a whole number g=m+1.
6. Process the particular multiplicative opposite: μ=λ- 1 mod m. General society (encryption) key is (n, m, g, e). The private (unscrambling) key is(d, x , y).

9. Diffie-hellman key exchange algorithm

The Diffie-Hellman algorithmic program is being employed to ascertain a shared secret which will be used for secret communications whereas exchanging knowledge over public network victimization the elliptic curve to get points and acquire the key victimization the parameters.

10.Project modules

This project contains several modules and described below.

11.Key Generation Module

In this module administrator going to produce two keys for encryption and decoding process. By utilizing Asymmetric calculation RSA, administrator going to produce asymmetric key and open a file.

Here the multilevel hashing method produces great and diverse hash esteems for various data sources. As shown in below figure we resolve the above issues with three key tables, LBA Remapping Table (LRT), Multi-index Table (MT), and Hash node Table (HT).

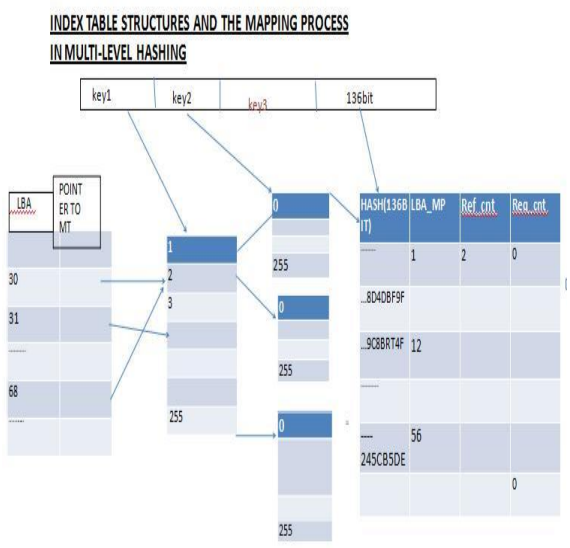

Fig.2. Multilevel hashing method.

We provide more details below. LRT is responsible for remapping the frontend I/O requests to the fingerprint index table. The first item of each entry in LRT records the LBA (logical block address) from the top of the file system, and the second item stores pointer information of the fingerprint table.

Considering the size of the fingerprint table, we present a novel structure, called MT, to reduce the table size. Instead of storing the full index table in RAM, MT is divided into many cells, called buckets, based on the first three 8 bits of the full fingerprint of the block. Every bucket has 256 entries with the same hash prefix.

As MT increases, all identical prefix buckets are organized with linked lists level by level. In order to save space with the HT, we use the remaining 136 bits as the key in HT in the last level. The design motivation of MT has another consideration, which also facilitates the cache of buckets using the high performance of SSD (solid state drive).

12.Access Control and Upload file using FTP protocol Module

In this module admin is going to give access control for the files upload. While uploading admin encrypts the file with the help of master secret key for the security purpose to the cloud using FTP protocol. Admin will be having the access control over the files hence achieves the security.

14.Intensive Data Migration.

At the point when new information things are embedded into capacity servers by means of cuckoo hashing, a kicking out task may cause concentrated information relocation among servers if hash impacts happen. The kicking-out activity needs to relocate the chose thing to its other applicant pails and kick out another current thing until the point that a vacant opening is found.Visit kicking-out activities cause serious information relocation among various containers of hash tables.

15.Send Aggregate Key

Based on the categories selected by admin, system has to fetch the corresponding hash keys and fetch the Public Key. Generate the User Aggregate Key and finally send it to users.

16.Search with Keyword Module

Users selects the aggregate key and provide the search keyword as input then the keyword will be converted into hash code then the aggregate will be decrypted, then the user gets the public key. Send the hash code to server after receiving the hash code the server has to check the keyword index and check for the matched files, if any found list the file names to the user. Then finally decrypt the file with the owner public key.

IV.CONCLUSION

This system developed using cuckoo hashing algorithm implemented and tested in hybrid cloud approach. The experimental results show the system

meet the all designed constraints which are discussed in system analysis.

The system automatically extract keywords from uploading file and the keywords are converted into hash key with respective access control while users are searching for a file by providing keywords based on users grade hash key will be generated and searched in hash key index and corresponding files are retrieved for security and efficiency.

V.FUTURE ENHANCEMENT

In the future this project aims in the following aspects

- More and more storage of files in the cloud.
- Reduces the hacking the files.
- Ensures more security.
- Fast access.

REFERENCES

[1]. Turner, J. Gantz, D. Reinsel, and S. Minton, "The digital universe of opportunities: Rich data and the increasing value of the internet of things," International Data Corporation, White Paper, IDC 1672, 2014.

[2]. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.

[3]. S. Bykov, A. Geller, G. Kliot, J. R. Larus, R. Pandya, and J. Thelin, "Orleans: cloud computing for everyone," Proc. SOCC, 2011.

[4]. S. Wu, F. Li, S. Mehrotra, and B. C. Ooi, "Query optimization for massively parallel data processing," Proc. SOCC, 2011.

[5]. Y. Hua, B. Xiao, and J. Wang, "Br-tree: a scalable prototype for supporting multiple queries of multidimensional data," Proc. TOC, vol. 58, no. 12, pp. 1585–1598, 2009.

[6]. C. Wang, K. Ren, S. Yu, and K. M. R. Urs, "Achieving usable and privacy-assured similarity search over outsourced cloud data," Proc. INFOCOM, pp. 451–459, 2012.

[7]. Q. Liu, C. C. Tan, J. Wu, and G. Wang, "Efficient information retrieval for ranked queries in cost-effective cloud environments," Proc. INFOCOM, pp. 2581–2585, 2012.

[8]. A. Crainiceanu, "Bloofi: a hierarchical bloom filter index with applications to distributed data provenance," the International Workshop on Cloud Intelligence, 2013.

[9]. N. Bruno, S. Jain, and J. Zhou, "Continuous cloud-scale query optimization and processing," VLDB Endowment, vol. 6, no. 11, pp. 961–972, 2013.

[10]. Y. Hua, B. Xiao, B. Veeravalli, and D. Feng, "Locality-sensitive bloom filter for approximate membership query," Proc. TOC, vol. 61, no. 6, pp. 817–830, 2012.

[11]. N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multikeyword ranked search over encrypted cloud data," Proc. TPDS, vol. 25, no. 1, pp. 222–233, 2014.

[12]. J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," Proc. INFOCOM, pp. 1– 5, 2010.

[13]. Y. Hua, B. Xiao, D. Feng, and B. Yu, "Bounded lsh for similarity search in peer-to-peer file systems," Proc. ICPP, pp. 644–651, 2008.

[14]. M. Bjorkqvist, L. Y. Chen, M. Vukoli ¨ c, and X. Zhang, "Minimizing ´ retrieval latency for content cloud," Proc. INFOCOM, pp. 1080–1088, 2011.

[15]. W. W. Peterson, "Addressing for random-access storage," IBM journal of Research and Development, vol. 1, no. 2, pp. 130–146, 1957.

[16]. J . I. Munro and P. Celis, "Techniques for collision resolution in hash tables with open addressing," ACM Fall joint computer conference, pp. 601–610, 1986.