JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Beneficial and Safe De-duplication for Block-Level Data in Public Clouds

[1]Veena M S, [2]Shobha Rani N R, [3]Revathi B

[1,2,3]Assistant Professor, Department of CSE, R R Institute of Technology, Bengaluru, Karnataka,

Abstract: Secure data deduplication can drastically reduce communication and storage overheads in cloud storage services in a society driven by big data. The bulk of data deduplication methods in use today either aim to withstand brute-force attacks or to ensure data efficiency and availability, but not both. Furthermore, to our knowledge, there is no existing project that encourages accountability by reducing the dissemination of unnecessary information (e.g., to determine whether plaintexts of two encrypted messages are identical). In this study, we investigate a three-tier cross-domain architecture and propose an efficient and privacy-preserving huge data deduplication in cloud storage (hereafter referred to as EPCDD). EPCDD safeguards privacy and guarantees data accessible while preventing brute-force attacks. Accountability is another factor we take into account while trying to offer better services. Then, we demonstrate that EPCDD outperforms currently employed rival techniques in terms of computing, communication, and storage overheads. The temporal complexity of the EPCDD duplication search is also logarithmic.

IndexTerms **– Deduplication, Security, Storage, Communication**

I. INTRODUCTION

While there is brute-force attack-resistant strategies in the literature, as far as we are aware, there isn't a single scheme that simultaneously offers effectiveness and data availability. Existing data deduplication systems cannot guarantee privacy with data encryption alone. For instance, duplicate information of the outsourced data left unprotected may have major privacy implications (e.g., to verify whether plaintexts of two encrypted communications are identical). Numerous occurrences have demonstrated that such information may invade people's privacy more than the primary data itself (e.g., NSA PRISM). However, current deduplication strategies make such information sharing predictable. As a result, we strive to minimise information leakage to the extent that only the party (the cloud storage provider) operating the deduplication is aware of it. Additionally, the cloud storage provider will be held responsible if the duplicate information leaks. It is obvious that it is difficult to create an effective deduplication method that achieves privacy preservation, availability, and accountability while fending off brute-force attacks. Therefore, we suggest an effective and privacy-preserving big data deduplication in cloud storage, referred to as EPCDD, in this study employing a three-tier cross-domain architecture. The EPCDD method successfully protects privacy while ensuring data availability, accountability, and resistance to brute-force attacks. To reduce the temporal complexity of duplicate search, we then build a deduplication decision tree based on the binary search tree. A dynamic tree, this deduplication decision tree supports data updates such data insertion, deletion, and modification.

1.Data de-duplication by maintaining only one copy of redundant information, deduplication is a technique for locating and removing duplicate data. In other words, data deduplication lowers the need for bandwidth as well as storage. However, the most common attack in safe data deduplication systems, brute-force attacks, pose a threat. As yet another effective secure deduplication strategy to stave off brute-force attacks. Although this method works well for deduplicating tiny data sets, it does not work well for deduplicating enormous amounts of data. To address this issue, Yan et al. suggested a technique for deduplicating encrypted massive data kept in the cloud using ownership challenges and proxy re-encryption. Despite being effective, this system is vulnerable to brute-force attacks.

2. EPCCD Effective privacy-preserving Data De-Duplication (EPCDD), which thwarts brute-force attacks, accomplishes privacy preservation, data availability, and accountability. We create a deduplication decision tree based on the binary search tree to reduce the time complexity of duplicate search. This deduplication decision tree supports data changes such data insertion, deletion, and modification.
Using infrastructure as dispensation chart conception ajar mound past-up availervices. Internet uses many entombment future more components to analyze the model of dimnerscutice. There endures many relevance for exemplification consider only three they inhabit many overhaul utility precedent spacious only three they are IAAS, PAAS and SAAS, keeping cost in mind go with it.

II. LITERATURE REVIEW

In order to save on storage space and related expenses, data deduplication techniques are being employed more frequently in cloud storage services like Dropbox, Google Drive, Mozy, and Sipderoak. The research community has recently researched secure data deduplication. In secure data deduplication, convergent encryption (CE), also known as content hash keying, is a cryptosystem that

creates identical ciphertexts from identical plaintext files. Efficiency is, generally speaking, a key indicator of whether a plan can be implemented in real life. Therefore, to utilise data deduplication in the real world, only securing data confidentiality is not enough. Based on their underlying architecture, secure data deduplication algorithms can also be divided into categories (i.e., client-side deduplication and server-side deduplication). Before it is sent, client-side deduplication alters the data at the client. As noted in, when building a deduplication scheme, dependability, security, and privacy should be taken into account in addition to attaining efficiency in storage, communication, and computing. While Zhou et al. proposed an effective secure deduplication scheme, which uses User-Aware Convergent Encryption (UACE) and Multi-Level Key management (MLK) approaches to fend off brute force attacks, Li et al. attempted to formalise the idea of distributed reliable deduplication system.

III. METHODOLOGIES

Users can check the accuracy of files stored remotely using dynamic proof of storage. Additionally, it offers a quick way to update those files when something changes. Large amounts of data can be stored in the cloud effectively with the help of dynamic PoS. Compared to single-user situations, multi-user environments pose particular difficulties. For instance, users may communicate with one another across a shared network connection, raising more security issues. Additionally, the system must be able to handle numerous transactions at once if multiple users are connected at once. The system must also be able to handle large numbers of transactions while still maintaining a low latency.

We must make sure that every data is encrypted before it is kept in the cloud in order to accomplish privacy preservation, accountability, and availability. In order to stop unauthorised users from accessing sensitive data, access restrictions must also be offered. We suggest a three-tiered cross domain design, consisting of a client tier, service tier, and database layer, to address these issues. Data must be encrypted on the client tier before being uploaded to the service tier. The service tier offers access controls and storage for encrypted data. The database layer, which comes last, houses the real data and offers the essential querying tools. Our system model consists of a Key Distribution Canter (KDC), Cloud Service Provider (CSP), Clients from various domains, and Local Managers. It is a three-tiered cross-domain large data de-duplication solution (LMs). We assume that while the CSP is honest but vengeful, the KDC is honest but curious. The CSP does not actively change saved messages because it has a pay-as-used business model. Because of this, we view the CSP as a passive foe.

IV. KEY GENERATION

KDC uses the composite bilinear parameter generator technique Gen () to generate a 5-tuple (N, g, G, GT, e) from a security parameter. Then, it chooses four random numbers at random from s, t, a, and b, where p | (as + bt), p, and p - bt, and computes yA = g aq G and yB = g bq G. KDC also selects three cryptographic hash functions, h1: 0, 1 0, 1 n, h2: 0, 1 Z p, and h3: G 0, 1 n, where n is the bit length of the symmetric key. Finally, the KDC delivers yA and yB to the CSP through the secure channel along with s and t to each member of domains A and B, respectively.

A. Data encryption and Tags generation Following receipt of the secret key(s), each client in domain A encrypts the data mi and creates relevant tags for data deduplication as shown below.

B. Data Encryption Using the secret key s and the parameter e(g, g) t, the client creates the message-dependent symmetric key ski = h1(mi||e(g, g) st). The client then chooses a random number, ri ZN, and computes the ciphertext, Ci = Encski (ri||mi), using the symmetric encryption algorithm, AES-CBC in the CBC mode

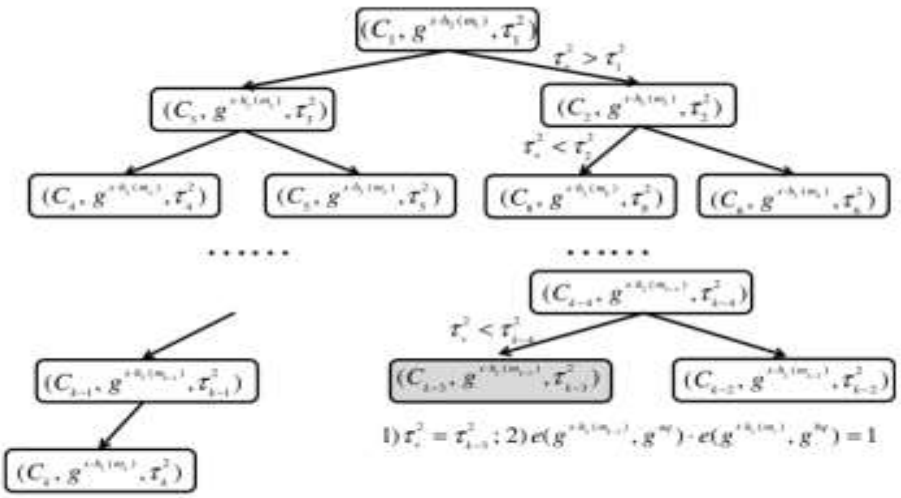


Figure1: Deduplication Decision Tree (DDT) Initialization To increase the effectiveness of finding duplicated data, we create deduplication decision trees (DDTs) based on the well-known binary search tree (BST).

C. Uploading data Let's imagine a client from domain B requests to upload data to the CSP. The LMB receives two tags from this client: 1 = g(tth2(m)) and 2 = h1(mke(g, g)st) mod. When LMB gets tags, it first calculates the hash value for 1, which is T = h3(1), and then looks up the hash value of the first tag by searching the hash table for all the different data from domain B. If the exact same hash value has already been stored, LMB informs the client of the duplicate find and does not need to send any messages to the CSP. The process for uploading data from domain A clients is identical to the process for uploading data from domain B clients, which is worth emphasising. Therefore, we fail to notice it.

V. SECURITY ANALYSIS

In this part, we examine the proposed EPCDD scheme's security attributes. Our research will pay special attention to outlining how the proposed EPCDD system may provide privacy preservation, data availability, and accountability while fending against brute-force attacks.

A. Privacy analysis We examine how well our EPCDD system can prevent the exposure of sensitive information while minimizing the disclosure of redundant information. Clients must not only upload the encrypted data but also two associated tags in order to work with the CSP to process data deduplication. The symmetric encryption algorithm, AESCBC, is used to encrypt Ci, hence the security of Ci is reliant on the symmetric encryption algorithm. Furthermore, dealing with the discrete logarithm (DL) problem and the one-way hash function, both of which have been demonstrated to be challenging problems that are computationally infeasible, is necessary if the CSP and LMA (LMB) are to acquire. Only CSP or LMA (LMB) can be used to solve an equation with two unknown values. Ski by assuming an assault. However, if |ski | = 256 bits, as explained in section 4.2, we can set || = 128 bits, which can effectively fend off the guessing attack. As a result, this paper can achieve data secrecy. Additionally, it must confirm that Eq. (1) holds in order to determine whether the various ciphertexts correspond to the same plaintext. Because only the CSP has access to the secret parameters g aq and g bq under our EPCDD scheme, only it is capable of conducting this verification. In other words, the duplicate information is only known to the CSP. As a result, our technique can minimise the disclosure of duplicate information.

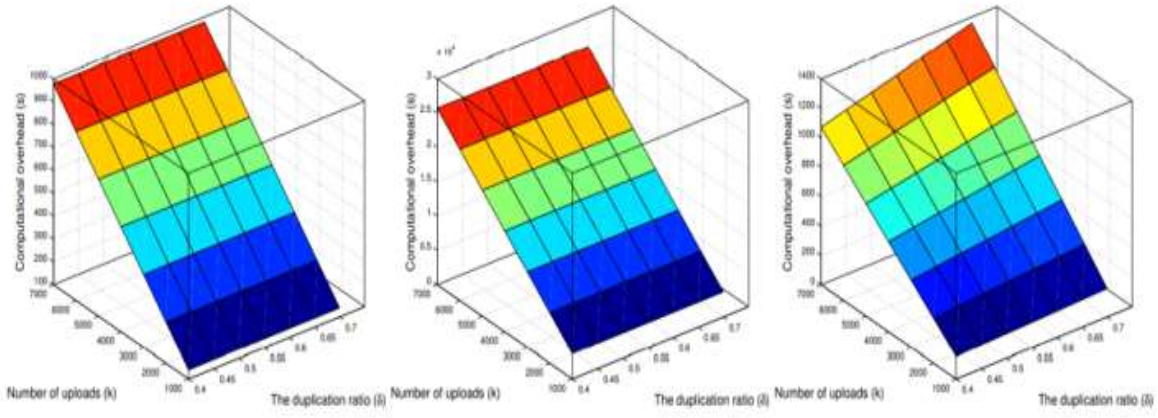
B. Brute-force Attack Resilience Although CSP has some knowledge of plaintext space M and maintains all clients' message tuples, our suggested EPCDD solution is still able to withstand brute-force attacks. In particular, despite having two secret parameters g aq and g bq, the DL problem's difficulty prevents the CSP from even obtaining aq or bq, much less s or t (p | as + bt). It is therefore challenging to determine e(g, g) st without s or t, which is the CDH problem, given e(g, g) s and e(g, g) t. Similarly, getting g is challenging. Although the CSP knows the plaintext space M and can attempt all data mi M for the particular message tuple, it is unable to produce the appropriate symmetric key ski = h1(mike(g, g)without e(g, g)st. As a result, without knowing ski and the random number ri, it is unable to decrypt Ci, much less produce the identical ciphertext Ci. is determined by ski, therefore without knowing ski, it cannot use to execute brute-force attacks. Additionally, CSP can determine the associated hashvalue, h2, for all mi M. (mi). It still is unable to compute, though. As a result, using brute-force methods, CSP is unable to determine if the plaintext and a particular message tuple are related. Regardless of the duplicated data that has been deleted, the client must make sure that this client can download and decrypt the stored ciphertext in order to access the data as long as the client has uploaded the ciphertext matching to the specific data. Let's use the example of client A from domain A needing to store mi. First, he communicates with the CSP by sending the message tuple (Ci, 1i, 2i). Then, CSP learns that Cithat has the same data that has already been stored. As a result, it is not required to store (Ci, 1i, 2i). Client A sends a request to download the encrypted data Ci after some time has passed, but the CSP only responds with Ci.

VI. PERFORMANCE EVALUATION

We assess the effectiveness of the suggested EPCDD method in terms of the overheads associated with computation, communication, and storage. Additionally, we provide a comparison with Yan's scheme and the R-MLE2 (Dynamic) scheme.

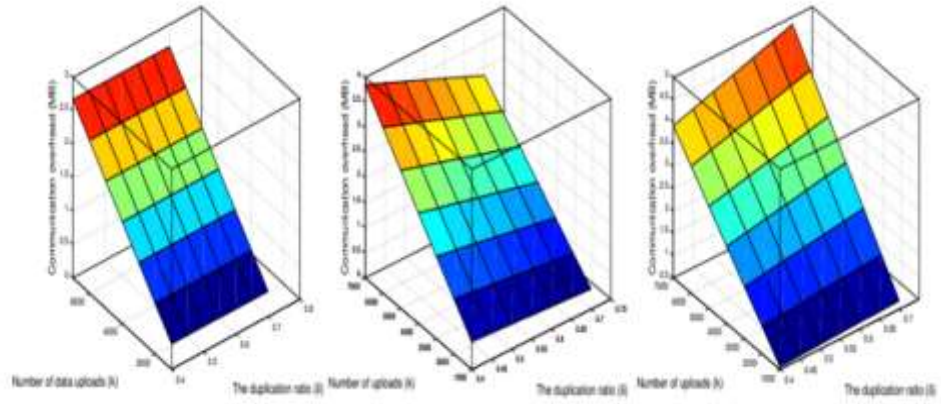
A. Computational Overheads In our EPCDD scheme, there are four different entities: customers, KDC, CSP, and LMA (LMB). According to the aforementioned system model, KDC is in charge of creating the system parameters but not the data deduplication. As a result, the KDC's computational overhead can be disregarded. We examine the computational cost of uploading a single data in two scenarios: duplicate data and fresh data.

B. Communication Overheads As previously stated, we do not include communication overheads for encrypted data Ci and instead discuss the overheads in two scenarios: one without duplication and one with duplication. Regardless of whether duplicate data exists, the client must send data to the CSP using the LMA or LMBits in our EPCDD scheme. We set 128 bits because the symmetric key for AESCBC is 256 bits long (n = 256 bits), which is more than enough for security. As a result, the message tuple has a size of 1152 bits. Consider that k data files from various customers must be uploaded. The duplication ratio in this case is, and the total communication overhead is 1152k bits.



(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

Figure2: A Comparative Summary: Computational Overheads

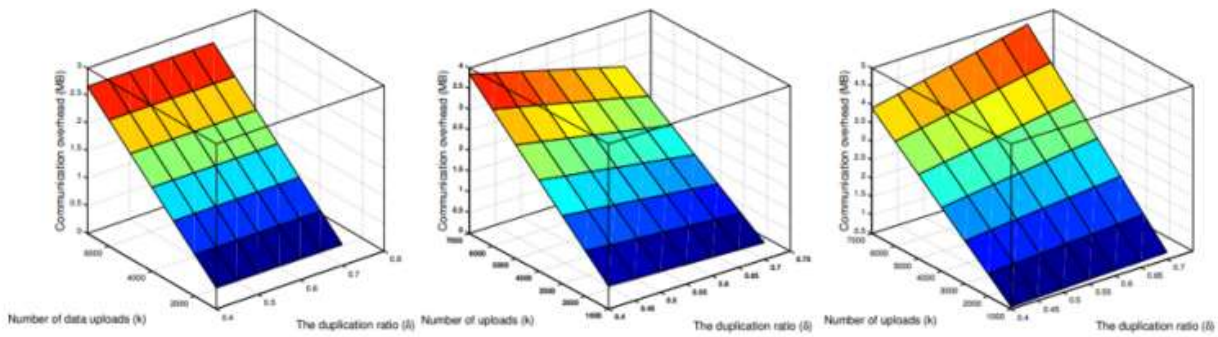


(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

Figure3: A Comparative Summary: Storage Costs

C. Storage Costs CSP doesn't keep any duplicate data messages. Therefore, CSP only has to store k(1) ciphertexts and the accompanying tags for k data with k duplicate data.

The storage costs of ciphertexts are also not included because they are the same for all three techniques. Thus, the storage costs of our EPCDD, R-MLE2 (Dynamic), and Yan's schemes are, respectively, 1152k(1), 6272k(1), and 3200k(1) bits. In terms of k and



(a) Our EPCDD scheme (b) μR-MLE2 (Dynamic) scheme [12] (c) Yan's scheme [14]

Figure4: A Comparative Summary: Communication Overheads

Compares the storage costs for these three strategies. We can see from the statistics that the storage costs for these three schemes rise as k rises and fall as rises. Furthermore, it is clear that our EPCDD solution saves storage costs in comparison to the competition.

VII. RESULTS

The bubble chart is another name for the DFD. It is a straightforward graphical formalism that may be used to depict a system in terms of the data that is fed into it, the different operations that are performed on it, and the data that is generated as a result of those operations. There were different system designs.

- Sequence Diagram (fig: sender user)

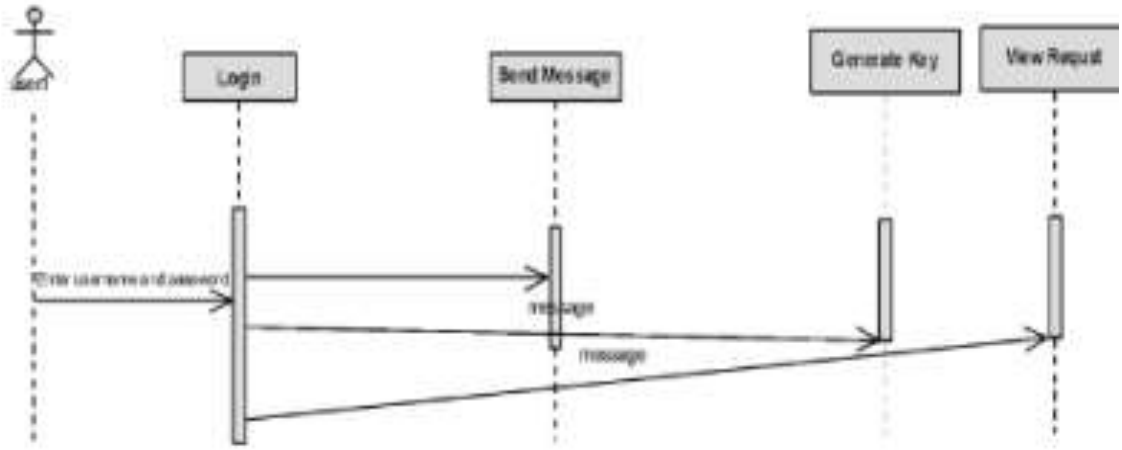


Figure5: Sender User Sequence diagram

The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Main Modules: -

1. User Module: Users in this module can access the information presented in the ontology system with authentication and security. Users must have an account in order to access or search the details; otherwise, they must register.

2. Secure Deduplication System: The tag of a file F will be decided by the file F and the privilege to support approved deduplication. We refer to it as a file token to highlight the differences between it and the conventional notation of tag. A file token will be produced using a secret key kp connected to a privilege p to support allowed access. The token of F that only a user with the privilege p is permitted to access is denoted by ′ F;p= TagGen(F, kp). In other words, only users with privilege p could compute the token ′ F;p. In another word, the token ϕ′ F;pcould only be computed by the users with privilege p. As a result, if a file has been uploaded by a user with a duplicate token ϕ′F;p, then a duplicate check sent from another user will be successful if and only if he also has the file F and privilege p. Such a token generation function could beeasily implemented as H(F, kp), where H(_) denotes a cryptographic hash function.

3. Security of Duplicate Check Token: We take into account a number of privacy types that we need to safeguard, namely: i the duplicate-check token's invulnerability. There are two different kinds of enemies: exterior enemies and inside enemies. The external enemy might be considered as an unprivileged internal adversary, as is demonstrated below. The adversary must be unable to create and print a legitimate duplicate token with any other privilege p′ on any file F, where p does not match p′, if a user has privilege p. It further stipulates that the attacker cannot fabricate and produce a legitimate duplicate token with p on any F that has been queried if it does not request a token with its own privilege from a private cloud server. 4. Send Key: After receiving a key request, the sender has the option of sending the key or declining it. The receiver can decrypt the message using this key and the request id that was produced at the time of sending the key request.

Figure6: Block Level De-Duplication

In Uploading file shows a new user can upload file may be txt,docs. The file will be accepted if it contains only unique values which are not present in storage.

Figure7: Data De-Duplication

Request Accept The represents request sent by the different user to the owner of the file to access the file that is present in server.

VIII. CONCLUSION

In the foreseeable future, it's anticipated that more people and businesses will continue to adopt cloud storage. This is due to the digitization of our society, which is not surprising. One related study area is how to effectively reduce cloud storage costs brought on by data duplication. In this study, we describe a three-tier cross domain architecture for huge data deduplication in cloud storage that is efficient and protects user privacy. The security of our suggested strategy was then investigated, and it was discovered that it improved data availability, accountability, and privacy preservation while thwarting brute-force attacks. We also showed that the proposed system beats current state-of-the-art methods in terms of computation, communication, and storage overheads. Additionally, our method's duplicate search time complexity is an effective logarithmic time.

REFERENCES

[1] R. Bhaskar, S. Guha, S. Laxman, and P. Naldurg, "Verito: A practical system for transparency and accountability in virtual economies," in 20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013, 2013. [Online]. Available: http://internetsociety.org/doc/ verito-practical-system-transparency-and-accountab ility-virtual-economies

[2] D. Boyd, K. Crawford, S. Shaikh, and V. Ravishankar, "Six provocations for big data," http://www.ils.albany.edu/wordpress/wp-content/up loads/ 2016/01/Six-provocations-for-Big-Data1.pdf.[3] Z. Yan, W. Ding, X. Yu, H. Zhu, and R. H. Deng, "Deduplication on encrypted big data in cloud," IEEE Trans. Big Data, vol. 2, no. 2, pp.138–150, 2016. [Online]. Available: http://dx.doi.org/10.1109/TBDATA.2016.2587659

[4] D. Boneh, E. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings, 2005, pp. 325–341. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30576-7 18

[5] "Openssl," https://www.openssl.org/.

[6] "Dropbox, a file-storage and sharing service," http://www.dropbox.com.

[7] "Google drive," http://drive.google.com.

[8] "Mozy: A file-storage and sharing service." http://mozy.com/.

[9] "Spideroak," https://www.spideroak.com/.

[10] D. T. Meyer and W. J. Bolosky, "A study of practical deduplication,"TOS, vol. 7, no. 4, p. 14, 2012. [Online]. Available: http://doi.acm.org/10.1145/2078861.2078864

[11] J. Li, X. Chen, X. Huang, S. Tang, Y. Xiang, M. M. Hassan, and A. Alelaiwi, "Secure distributed deduplication systems with improved reliability," IEEE Trans. Computers, vol. 64, no. 12, pp. 3569–3579,2015. [Online]. Available: http://dx.doi.org/10.1109/TC.2015.2401017 [15]Z. Yan, W. Ding, X. Yu, H. Zhu, and R. H. Deng, "Deduplication on encrypted big data in cloud," IEEE Trans. Big Data, vol. 2, no. 2, pp.138–150, 2016. [Online]. Available: http://dx.doi.org/10.1109/TBDATA.2016.2587659

[13] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, "Secure and efficient cloud data deduplication with randomized tag," IEEE Trans. Information Forensics and Security, vol. PP, no. 99, pp.1–1,2016.[Online].Available:http://dx.doi.org/10 .1109/TIFS.2016.2622013

[14] Basu, S. 1997. The Investment Performance of Common Stocks in Relation to their Price to Earnings Ratio: A Test of the Efficient Markets Hypothesis. Journal of Finance, 33(3): 663-682.

[15] Bhatti, U. and Hanif. M. 2010. Validity of Capital Assets Pricing Model.Evidence from KSE-Pakistan.European Journal of Economics, Finance and Administrative Science, 3 (20).

[16] R Manjunath, A Akshatha, S Balaji, "Reverse engineering in Big Data using Cloud computing and Open Stack virtual machine", International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 848-853, 2016.