International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

Achieving Efficient and Privacy-Preserving Cross-Domain Big Data Deduplication in Cloud

P. Salomi Priya[1], Dr. S. Christy Melwyn[2], Dr. K. Ravi Kumar[3]
[1, 2, 3]*Rrase College of Engineering, Padappi-601301.*

Abstract: ***Secure data deduplication can significantly reduce communication and storage overheads in cloud storage services, and has potential applications in our big data-driven society. In Existing, there are several problems arises to comparing files in an efficient way. That matches by filename, not its content. So here if filename different and content is same so here duplication file will occur continuously. Second thing is each file will upload by user it will read and existing file character by character, So here the time efficiency is main problem to compare both files. In database consists billions and trillions of files there. If this comparison may persist the will increased continuously. In Proposed, we introduce Hash function, this function greatly helps for us to comparing every file. Initially when we upload files, hash code will generate for each file and store into hash table and the file will encrypted with the private key and store to the database. When another user uploads a same file, the same step begins to generate the hash function and now this hash function will compare to the hash table. When the hash matches the file will blocked otherwise it will stored. . Here we reduce lot of time compare to existing. Filename matches problem will not occur. Here we make a better privacy too. EPCDD achieves both privacy-preserving and data availability, and resists brute-force attacks. In addition, we take accountability into consideration to offer better privacy assurances than existing schemes.***
Keyterms: Secure data deduplication, big data, brute-force attacks, data availability, accountability.

I. INTRODUCTION

Data deduplication technique has increasingly been used in cloud storage services, such as Dropbox, Google Drive, Mozy, Sipderoak, to reduce storage space and the associated costs. Such technique can be broadly categorized based on the level of granularity, namely: file-level and chunk level. We refer interested reader for a performance comparison between file-level and chunk-level deduplication approaches. Existing data deduplication schemes are generally designed to either resist brute-force attacks. Data encryption alone is insufficient to ensure privacy in existing data deduplication schemes. For example, duplicate information (e.g., to determine whether plaintexts of two encrypted messages are identical) of the outsourced data left unprotected may have serious privacy implications. Suppose the CSP has some background knowledge of the plaintext space. Although the CSP stores encrypted message tuples of all clients and has knowledge of some secret parameters received from the KDC, it is not able to obtain the plaintext corresponding to the specific ciphertext through brute-force attacks. CLOUD storage usage is likely to increase in our big data driven society. For example, IDC predicts that the amount of digital data will reach 44 ZB in 2020. Other studies have also suggested that about 75% of digital data are identical (or duplicate), and data redundancy in backup and archival storage system is significantly more than 90%. While cost of storage is relatively cheap and advances in cloud storage solutions allow us to store increasing amount of data, there are associated costs for the management, maintenance, processing and handling of such big data. It is, therefore, unsurprising that efforts have been made to reduce overheads due to data duplication. The technique of data deduplication is designed to identify and eliminate duplicate data, by storing only a single copy of redundant data. In other words, data deduplication technique can significantly reduce storage and bandwidth requirements. However, since users and data owners may not fully trust cloud storage providers, data (particularly sensitive data) are likely to be encrypted prior to outsourcing. This complicates data deduplication efforts, as identical data encrypted by different users (or even the same user using different keys) will result in different cipher texts. Thus, how to efficiently perform data deduplication on encrypted data is a topic of ongoing research interest. Cloud computing is clear that designing an efficient deduplication scheme that achieves privacy-preserving, availability and accountability, while resisting brute-force attacks remains challenging. Therefore, in this paper, using three-tier cross-domain architecture, we propose an efficient and privacy-preserving big data deduplication in cloud storage, hereafter referred to as EPCDD. The EPCDD scheme achieves privacy-preserving, data availability and accountability, as well as resisting brute-force attacks. We then construct a deduplication decision tree based on the binary search tree to improve the time complexity of duplicate search. This deduplication decision tree is a dynamic tree that supports data update such as data insertion, deletion and modification. The main scope of the project is avoid the duplicate in cloud. It is also reduce the time with secure.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

II. PROPOSED METHOD

A. Information Collection

Suppose a client in domain B wishes to upload the data to the CSP. This client computes two tags and sends them to the LMB. Upon receiving tags, LMB computes the hash value for , and then searches the hash table that records hash values of the first tag for all different data from domain B. If the same hash value has already been recorded, LMB returns this client "duplication find", and does not need to forward any message to the CSP. Otherwise, LMB sends the tags to the CSP. After receiving it, CSP checks the duplication on the DDT-A. If the duplicated data exist, CSP sends "duplication find" to the LMB. Otherwise, it sends "upload data" to the LMB. After receiving the feedback, LMB forwards it to the client. Once receiving the message "upload data", the client encrypts and then sends it to the CSP via the LMB. After receiving the cipher text CSP leverages Algorithm to insert the message tuple into the appropriate node in the DDT-B. It is worth noting that the process of uploading data from clients in domain A is identical to B. Hence, we ignore this process.

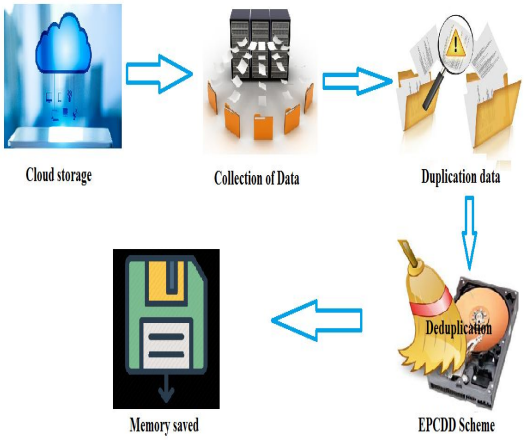

Figure 2.1 Example of EPCDD architecture diagram.

B. File upload and Encryption

CSP has some background knowledge of plaintext space M, and stores the message tuples of all clients, encryption is the process of encoding a message or information in such a way that only authorized parties can access it and those who are not authorized cannot. Encryption does not itself prevent interference, but denies the intelligible content to a would-be interceptor. In an encryption scheme, the intended information or message, referred to as plaintext, is encrypted using an encryption algorithm a cipher generating cipher text that can only be read if decrypted. For technical reasons, an encryption scheme usually uses a psedo random encryption key generated by an algorithm. It is in principle possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, considerable computational resources and skills are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients but not to unauthorized users.

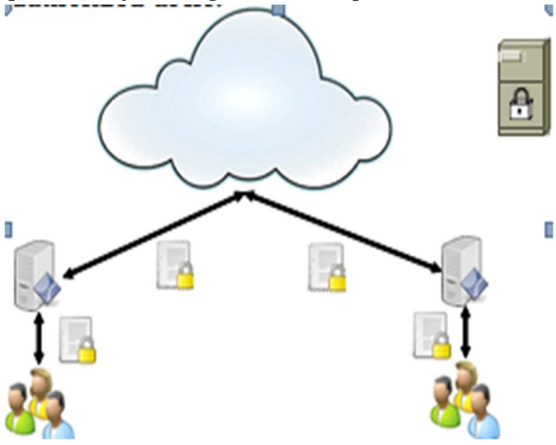

Figure .2.2 System model under consideration

C. *Information Manager*

We analyze that our EPCDD scheme can protect the privacy of sensitive data from disclosure, and minimize the duplicate information disclosure. In order to cooperate with the CSP to process data deduplication, clients need to not only upload the encrypted data, but also provide two corresponding tags. Because Ci is encrypted by the symmetric encryption algorithm, i.e., AESCBC, the security of Ci is based on the symmetric encryption algorithm. Moreover, if the CSP and LMA (LMB) want to obtain mi from the tag it means that they need to deal with the discrete logarithm (DL) problem and the one-way hash function, which have been proved to be computationally infeasible difficult problems. Therefore, CSP or LMA (LMB) cannot obtain mi from the there is an integer k One equation has two unknown numbers, CSP or LMA (LMB) only can obtain ski by guessing attack, which can sufficiently resist the guessing attack. Therefore, data confidentiality can be achieved in this paper. In addition, to verify whether the different ciphertexts correspond to the identical plaintext, it needs to verify holds. As designed in our EPCDD scheme, only the CSP has the secret parameters gaq and gbq, so that only it can perform this verification. In other words, only the CSP knows the duplicate information. Thus, Our scheme **can** reduce the disclosure of duplicate information as much as possible.

D. *Verify Deduplication*

We propose an efficient and privacy-preserving cross-domain deduplication scheme for big data storage (EPCDD). In order to improve the efficiency of finding duplicated data, we construct the deduplication decision trees (DDTs) based on the popular binary search tree (BST) for searching duplicate data. As far as we know, the DDT initialization is similar to insertion of the BST, but this operation begins with the empty tree. Suppose the CSP has received k different message tuples form domain A at the current moment. Hence, CSP constructs a DDT-A for this domain to store k message tuples for subsequence deduplication. Based on the insertion operation of the BST, we propose Algorithm to construct DDTs. According to Algorithm, CSP stores k message tuples in turn at appropriate nodes. In addition, in order to ensure the time complexity of searching duplicate is O(log k), we need to balance the tree in the process of the DDT construction.

III. MODELS AND DESIGN GOALS 3.1System Model

The system model (see Fig.1) is a three-tier cross-domain big data deduplication system, which comprises a key distribution center (KDC), a cloud service provider (CSP), clients from different domains and the corresponding local managers, denoted as LMA and LMB.

1) *KDC:* The trusted KDC is tasked with the distribution and management of private keys for the system.
2) *CSP:* The first tier is a CSP, which offers data storage services for clients. While the CSP is capable of supporting the storage needs of clients, it is financially vested to reduce the expensive big data management and maintenance overheads. Therefore, the CSP needs to perform inter-deduplication, which means that messages for deduplication are from different domains, to decrease the corresponding overhead.
3) *Clients:* Every client is affiliated with a domain (e.g., employees in the company or students and faculty members in the university or university network, say University of Texas system). Clients upload and save their data with the CSP. In order to protect their data privacy and help the CSP to complete data deduplication over encrypted data, they encrypt the data and generate the corresponding tags.

A. *Threat Model*

In our threat model, the CSP is considered honest but curious, which is the most common assumption in the literature (see [12], [14], [18]). Specifically, the CSP honestly follows the underlying scheme.

However, it is curious about contents of stored data. Because the CSP adopts a pay-as-you-use model, it does not actively modify stored messages due to reputation, financial and legal implications (e.g. a civil litigation can result in significant reputation and financial losses to the provider).

Hence, active attacks from the CSP are not considered in this paper. However, due to the significant amount of data stored in the cloud, it may know the plaintext space. Hence, according to the ciphertext and corresponding tags, the CSP (e.g. a malicious CSP employee) can carry out brute-force attacks. Finally, the CSP may obtain the plaintext corresponding to the special ciphertext for other illicit purposes (e.g. information reselling for financial gains).

LMA and LMB are also considered honest but curious.

However, these entities have very limited computing and storage capabilities. Therefore, in practice, they do not have sufficient resources to carry out brute-force attacks. LMA or LMB may be curious about its affiliated clients' privacy, even though they may

not actively seek to compromise the privacy of their clients. For example, if the domain is a company and LMA (or LMB) is the corresponding information manager. LMA (or LMB) is curious about the data uploaded by the staff. However, to protect the information asset, LMA or LMB does not actively attempt to compromise the privacy of clients, or collude with the CSP.
Clients are considered honest.
In theory, it is possible that they would collude with the CSP to obtain other clients' privacy. As mentioned in [14], in practice, such collusion may result in significant risks to the reputation of the CSP, as well as civil litigation or criminal investigations. In addition, if the CSP colludes with client A to compromise the privacy of client B, the CSP is also likely to collude with client B or other clients to compromise the privacy of other existing clients. This would have serious repercussions for the CSP if such collusion is reported or known. Thus, we assume that the CSP does not collude with its clients. Other than brute-force attacks, we do not consider other active attacks.

IV. PROPOSED EPCDD SCHEME

In this section, we propose an efficient and privacy-preserving cross-domain deduplication scheme for big data storage (EPCDD).

A. Key Generation

KDC takes a security parameter κ as input, and outputs a 5-tuple (N, g, G, G_T , e) by running the composite bilinear parameter generator algorithm G$en(\kappa)$. Then, it selects four random numbers $s, t, a, b \in Z_N$, where $p \mid (as + bt)$, $p \ddagger as$ and $p \ddagger bt$, and computes $y_A = g^{aq} \in G$, $y_B = g^{bq} \in G$. In addition, KDC chooses three cryptographic hash functions $h : \{0, 1\}^* \rightarrow \{0,1\}^n$,

B. Data Encryption and Tags Generation

For each client in domain A, after receiving the secret key s, the client encrypts the data m_i and generates corresponding tags for data deduplication as follows. Similarly, clients in domain B execute same operations to generate ciphertexts and tags, e.g., for m_j, $C_j = Enc_{sk}(r_j \| m_j)$, where $sk_j = h_1(m_j \| e(g, g)^{st})$, and the corresponding tags are computed as $\tau^2 = sk_j \bmod \omega$.

C. Deduplication Decision Tree (DDT) Initialization

Based on the insertion operation of the *BST*, we propose Algorithm 1 to construct *DDT*s. According to Algorithm 1, CSP stores k message tuples in turn at appropriate nodes, as shown in Fig.2. In addition, in order to ensure the time complexity of searching duplicate is $O(\log k)$, we need to balance the tree in the process of the *DDT* construction.
We obtain:

$$e(\tau^1, g^{aq}) \cdot e(\tau^1, g^{bq})$$
$$= e(g^{s \cdot h2(m_i)}, g^{aq}) \cdot e(g^{t \cdot h2(m_j)}, g^{bq})$$
$$= e(g, g)^{asqh2(m_i)+btqh2(m_j)} \quad (3)$$
$$= e(g, g)^{(as+bt)q \cdot h2(m_i)}$$
$$\because as+bt=kp,\ e(g,g)^N = 1$$
$$= e(g, g)^{kN \cdot h2(m_i)} = 1 \quad (4)$$

Hence, when $m_i = m_j$, Eq. (2) always holds.

V. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed EPCDD scheme in terms of the computational, communication and storage overheads. Moreover, we give a comparison with the μR-MLE2 (Dynamic) scheme [12] and Yan's scheme [14].

A. Computational Overheads

There are four entities in our EPCDD scheme, namely: clients, KDC, CSP and LMA (LMB). Under the aforementioned system model, KDC is responsible for generation of system parameters, which does not participate in the data deduplication. Thus, the computational overhead of the KDC can be ignored. We analyze the computational overhead of uploading one data in two cases: the data is duplicate and the data is new.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

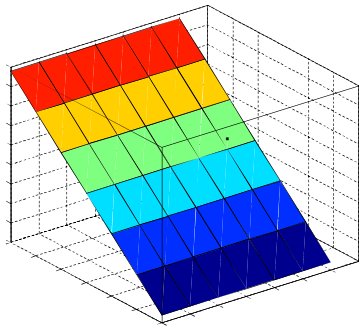

Figure.5.1 A comparative summary: computational overheads

B. Communication Overheads

As described above, we omit communication overheads of en- crypted data C_i and discuss the overheads in two cases: without the duplication and with the duplication. In our EPCDD scheme, regardless of whether duplicated data exist, the client needs to send τ^1 τ^2 to the CSP via the LMA or LMB, which costs $\tau^1 + \tau^2$ bits. Because the length of symmetric key for *AES- CBC* is 256 bits (n = 256 bits), we set ω = 128 bits, which is sufficient for the security of τ^2. Thus, the size of the message tuple is $\tau^1 + \tau^2 = 1152$ bits. Consider k data from different clients need to be uploaded, wherein the duplication ratio is δ, the whole communication overheads are $1152k$ bits. In addition, KDC needs to send secret parameters to k clients and the CSP, the messages are in the form of s $e(g, g)^t$ $(t$ $e(g, g)^s)$ and y_A y_B, respectively. Thus, its size should be $2048k$ and 2048 bits, respectively. In summary, the total communication overheads of our EPCDD are $(3200\ k + 2048)$ bits.

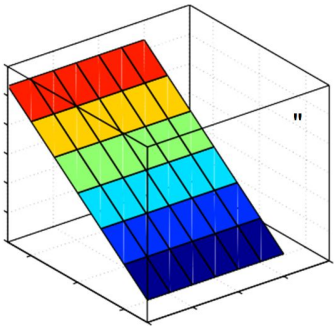

Figure.5.2. A comparative summary: communication overheads

C. Storage Costs

As mentioned above, CSP does not store any message of duplicated data. Thus, for k data with $k\delta$ duplicate data, CSP just needs to store $k(1\ \delta)$ ciphertexts and the corresponding tags. Similarly, we omit the storage costs of ciphertexts as the same part in the three schemes. Hence, the storage costs of our EPCDD, μR-MLE2 (Dynamic) and Yan's schemes are $1152k(1\ \delta)$ bits, $6272k(1\ \delta)$ bits and $3200k(1\ \delta)$ bits, respectively. Fig shows the comparison of storage costs for these three schemes in terms of k and δ. From the figures, we can see that the storage costs for these three schemes increase as k increases and decrease as δ increases.

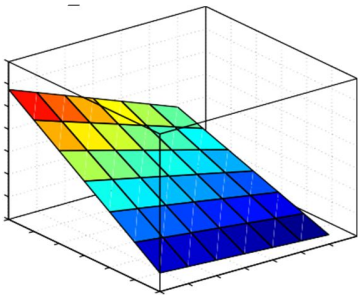

Fig.5.3. Evaluation of storage cost.

VI. CONCLUSION

Future research includes extending the proposed scheme to fully protect the duplicate information from disclosure, even by a malicious CSP, without affecting the capability to perform data deduplication. We then analyzed the security of our pro- posed scheme and demonstrated that it achieves improved privacy- preserving, accountability and data availability, while resisting brute-force attacks. We also demonstrated that the proposed scheme outperforms existing state-of-the-art schemes, in terms of computation, communication and storage overheads. Future research agenda will also include extending the scheme to be resilient against a wider range of security threats by external attackers, as well as improving the time complexity of duplicate search.

REFERENCES

[1] J.Hu, R. Marculescu, DyAD – smart routing for networks-on-chip, in: 41st Design Automation Conference, 2004, pp. 260–263.
[2] M. Li, Q. Zeng, W. Jone, DyXY – a proximity congestion-aware deadlock-free dynamic routing method for network on chip, in: 43rd ACM/IEEE Design Automation Conference, 2006, pp. 849–852.
[3] B. Niazmand, M. Reshadi, A. Reza, PathAware: a contention-aware selection function for application-specific Network-on-Chips, in: NORCHIP, 2012, pp. 1–6
[4] G. Ascia, V. Catania, M. Palesi, D. Patti, Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip, IEEE Trans. Comput. 57 (6) (2008) 809–820.
[5] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, B. McGinley, F. Morgan, Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive Network-on-Chip routers, Neural Networks 33 (9) (2012) 42–57.
[6] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, B. McGinley, F. Morgan, Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive Network-on-Chip routers, Neural Networks 33 (9) (2012) 42–57.
[7] P. Lotfi-Kamran, A.M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, Z. Navabi, EDXY – a low cost congestion-aware routing algorithm for Network-on Chips, J. Syst. Arch. 56 (7) (2010) 256–264.
[8] X. Chang, M. Ebrahimi, M. Daneshtalab, T. Westerlund, J. Plosila, PARS – an efficient congestion-aware routing method for networks-on-chip, in: 16th CSI International Symposium on Computer Architecture and Digital Systems, 2012, pp. ng, W. Jone, DyXY – a proximity congestion-aware deadlock-free dynamic routing method for network on chip, in: 43rd ACM/IEEE Design Automation Conference, 2006, pp. 849–852.
[9] B. Niazmand, M. Reshadi, A. Reza, PathAware: a contention-aware selection function for application-specific Network-on-Chips, in: NORCHIP, 2012, pp. 1–6
[10] G. Ascia, V. Catania, M. Palesi, D. Patti, Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip, IEEE Trans. Comput. 57 (6) (2008) 809–820.
[11] S. Carrillo, J. Harkin, L. McDaid, S. Pande, S. Cawley, B. McGinley, F. Morgan, Advancing interconnect density for spiking neural network hardware implementations using traffic-aware adaptive Network-on-Chip routers, Neural Networks 33 (9) (2012) 42–57.
[12] P. Lotfi-Kamran, A.M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, Z. Navabi, EDXY – a low cost congestion-aware routing algorithm for Network-on Chips, J. Syst. Arch. 56 (7) (2010) 256–264.
[13] X. Chang, M. Ebrahimi, M. Daneshtalab, T. Westerlund, J. Plosila, PARS – an efficient congestion-aware routing method for networks-on-chip, in: 16th CSI International Symposium on Computer Architecture and Digital Systems, 2012, pp. 166–171.
[14] Z. Lu, M. Zhong, A. Jantsch, Evaluation of on-chip networks using deflection routing, in: 16th ACM Great Lakes Symposium on VLSI, 2006, pp. 296–301.
[15] N. Jiang, J. Kim, W.J. Dally, Indirect adaptive routing on large scale interconnection networks, in: 36th Annual International Symposium on Computer Architecture, 2009, pp. 220–231.
[16] R. Manevich, I. Cidon, A. Kolodny, I. Walter, Centralized adaptive routing for NoCs, IEEE Computer. Arch. Lett. 9 (2) (2010) 57–60.