JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 36, 993-1005 (2020)
DOI: 10.6688/JISE.202009_36(5).0004

An Integrated Approach for Data Leaker Detection in Cloud Environment

ISHU GUPTA+ AND ASHUTOSH KUMAR SINGH
Department of Computer Applications
National Institute of Technology
Kurukshetra, 136119 India
E-mail: ishugupta23@gmail.com; ashutosh@nitkkr.ac.in

Data leakage is one of the critical challenges in the area of cloud computing where data sharing is an essential part among multiple entities. This work presents a generic Data Leaker Detection Model (DLDM) that identifies the malicious entity responsible for data leakage. The proposed approach is an integration of cryptography, watermarking and hashing techniques for securing the data. Furthermore, the model detects the venomous user by considering a combination of watermark extraction and probability estimation. The results signify that the time taken to detect the malicious user is 3580 ms when 200 documents of size 20 MB are provided to single/distinct users. The average probability to identify the venomous user is 0.969518 for the load value of 2, which indicates the high probability to identify the guilty agent. The experimental results verify the efficiency of the proposed model.

Keywords: cloud computing, data leakage, data security, guilty client, probability

1. INTRODUCTION

A substantial number of organizations are shifting to the cloud due to its several characteristics such as scalability, robustness, and on-demand services, *etc*. The critical organizational data stored in the cloud need to be shared among various supposedly trusted parties within or outside the organizational premises in the context of persisting the businesses. Sharing of cloud data among multiple entities comprises a number of threats to the organization since it can be leaked by any indignant entity to the unauthorized third party [1, 2]. According to an exotic chronology of data breaches explored by Privacy Rights Clearinghouse (PRC), $11,587,118,203$ records have been breached in the United States alone from $9,002$ data breaches made public since 2005 [3]. Leakage of sensitive data in deliberate or undeliberate manner by the malicious external entities or the indignant internal entity poses one of the most grievous security threats to the organization's confidentiality and individual's privacy. According to 2018 annual cost of data breach study conducted by Ponemon institute, the average consolidated cost of a data breach raised from \$3.62 million to \$3.86 million in the year 2018, which implies an increase of 6.4%

Received July 30, 2019; accepted October 4, 2019.
Communicated by Huo Chong Ling.
+ Corresponding author.

over the previous year. The cost procured for each stolen or lost record consisting confidential information has increased by 4.8% from consolidated average of $141 to $148 [4]. Because of these reasons, data leakage problem has become a critical challenge in the cloud computing and it keeps on increasing as number of cloud users are rising [5, 6]. Thus, it has become necessity to protect the sensitive data from any unauthorized access. Several work has been reported to address this problem which can be broadly categorized in cryptography, watermarking, and probability based methods.

1.1 Key Contribution

A user centric key management scheme to protect the cloud data is given by Kao *et al*. in [7] where the authors used RSA encryption technique to encrypt the user data. Al-Haj *et al*. [8] introduced a strong cryptographic function by using hash code and symmetric keys to provide the confidentiality, integrity and authenticity to the data. A ciphertext policy attribute based non circuit homomorphic encryption scheme is presented by Tan *et al*. in [9] for provisioning the fine-grained access control to the cloud data which is outsourced among multi-user Pandiaraja *et al*. [10] proposed an attribute based scheme for securely accessing the data stored in the public cloud. The scheme managed the broadcast key by carrying out minimum number of cryptographic operations. A revocable-storage identity-based encryption scheme is provided by Wei *et al*. in [1] that performed the ciphertext updation and user revocation concurrently to provide the security while sharing the cloud data. Despite of the robustness of the cryptography method for securing the sensitive data, this method has a major drawback that the data can be compromised once the key used for encryption is cracked. Furthermore, the method is incapable to detect the malicious entity when the data has leaked.

Bishop *et al*. presented a mobile agent based approach in [11] that brutalizes the process of discovering and coloring perceptive hosts file systems and observing the colored file system for detection of potential information leakage to unauthorized third party. A model that identifies the guilty party who is responsible for leaking cloud data via utilizing watermarking technique and Bell La Padula Model is given by Kumar *et al*. in [6]. Backes *et al*. [12] provided a framework called LIME to identify the guilty entity by developing a data transfer protocol between two entities via considering a combination of oblivious transfer, watermarking and signature primitives techniques. The cloud data is protected by Shen *et al*. in [2] via hiding the critical data stored in the cloud when it is shared among the users. A layered based approach which is an integration of hashing, cryptography, and watermarking techniques for cloud data security and vicious user detection is presented in [13]. Furthermore, a data leaker detection model is presented in [14] for securely sharing the data stored in the cloud that assured the data confidentiality. Although, the watermarking is an efficient technique to find the client who leaked the data, but it fails when the embedded code is modified or completely destroyed by the client.

Papadmitriou *et al*. [15] proposed a probabilistic method to assess the guilt of various agents which consists various algorithms for the distribution of data among multiple agents. Kumar *et al*. [16] introduces the allocation strategies that works on the basis of no wait model and increases the chances of identifying the guilty party. Sodagudi *et al*. proposed an approach in [17] to identify the malicious attackers in mobile ad-hoc

network (MANET) via considering a combination of routing protocol and cryptography technique. A guilty agent detection model is reported that considered the sample data for the distribution among the agents and the guilty entity is estimated based on the allocated data represented through bigraph [18]. Moreover, a threshold based scheme is proposed in [5] that recognized the guilty party causing the leakage of the data by handling the explicit data request where the prescribed objects are requested by the agents. The advantage of probabilistic method is that the detection of guilty entity is not affected by alteration/destruction in the data and the key acquisition unlike watermarking and cryptography technique respectively. But, this method provides the estimation only of the guilty user which becomes difficult when the overlapping of data among the agents increases.

1.2 Our Contributions

In order to improve the security for sharing the critical information in the cloud environment, a Data Leaker Detection Model (DLDM) is proposed that addresses the data leakage problem and preserves the data confidentiality by detecting the malicious entity. For this purpose, the model utilized watermarking and probabilistic approaches. Furthermore, the hashing technique for authentication purpose and cryptography technique to provide stronger security to the protocol are employed. Our contribution includes: (1) deals with an untrusted entity associated with transfer of data among three party system (2) secures the confidential information being transmitted and protect it from unauthorized use (3) identifies the guilty entity responsible for unknowingly or advertantly leaking the critical data to an unauthorized third party

The rest of the paper is organized as follows: The proposed model DLDM is discussed in Section 2. Section 3 describes the embedding scheme for the purpose of hiding the secret information. Section 4 presents the identification of guilty entity followed by performance evaluation and conclusions in Sections 5 and 6 respectively.

2. DATA LEAKER DETECTION MODEL

The DLDM consists three different entities that are data owners $\mathbb{O} = \{O_1, O_2, \ldots, O_q\}$, cloud $\mathbb{CL} = \{Cl_1, Cl_2, \ldots, Cl_p\}$ and the clients $\mathbb{C} = \{C_1, C_2, \ldots, C_m\}$. The data owner $O_k; k \in \{1,2,\ldots,q\}$ owns the valuable data $\mathbb{D} = \{D_1, D_2, \ldots, D_n\}$ where $D_i; i \in \{1,2,\ldots,n\}$ are independent data objects of various forms. $O_k \in \mathbb{O}$ uploads the data $D_i \in \mathbb{D}$ on cloud $Cl_h; h \in \{1,2,\ldots,p\}$ that are demanded by various clients $C_j \in \mathbb{C}$. The master server S_v provides the requested data set $Z_j \subseteq \mathbb{D}$ to various $C_j; j \in \{1,2,\ldots,m\}$. As the data is shared among three party system, therefore data confidentiality is addressed as the most serious hazard in the model. The entities O_k and Cl_h are considered to be honest. The honesty means that $\mathbb{D}$ is not revealed by these two parties. It is believed that O_k is primarily concerns about the data security and it is affected the most by the data leakage, hence it can't leak its own data $\mathbb{D}$. Furthermore, it is investigated that the data access policies are honestly followed by $Cl_h \in \mathbb{CL}$, therefore, it is taken into account that Cl_h also can't expose $\mathbb{D}$. The entity $C_j \in \mathbb{C}$ is regarded as an untrusted party in the model. Although the data $\mathbb{D}$ is provided to authorized C_j only, but it can't be ascertained that no C_j will disclose the data $D_i \in \mathbb{D}$ after receiving it. On receiving the data set Z_j, if any

client $C_j \in \mathbb{C}$ leaks at least one data object D_i from its allocated data set Z_j with target t, this entity is named as guilty client (G_c). The proposed approach provides a mechanism to detect G_c even though C_j tries to tamper the shared document.

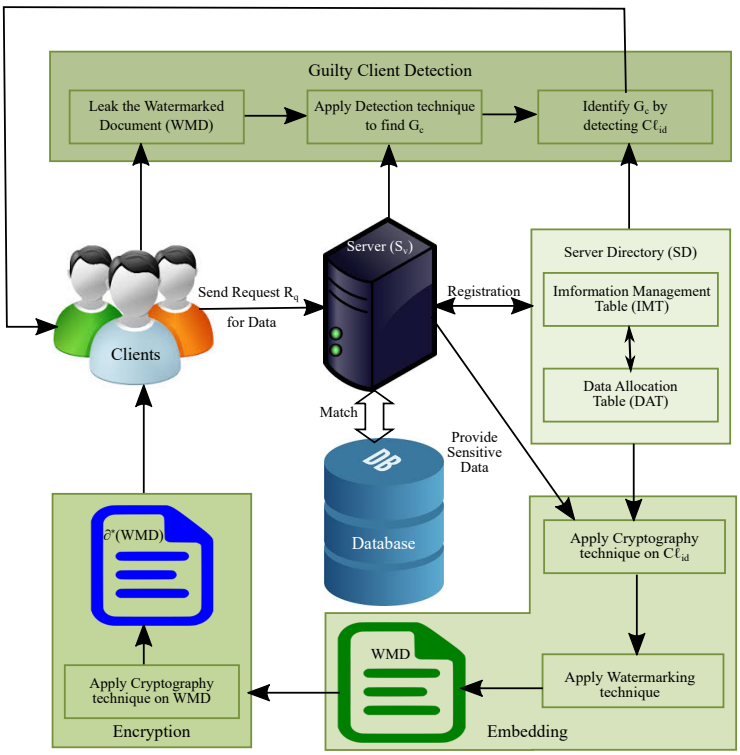


Fig. 1. Proposed architecture DLDM.

Fig. 1 depicts the conceptual framework of the proposed architecture initiating from data sharing to the guilty client detection. Client (C_j) sends the request (R_q) to the server (S_v) for required document (D_i) stored in the cloud (Cl_{id}). If D_i is available in database (D_b) then S_v checks Information Management Table (IMT) to find whether C_j has already registered or not. Otherwise C_j has to register with S_v then $C\ell_{id}$ is generated by S_v for C_j. After that S_v inserts the logo of the organization L_o to which C_j belongs in D_i for embedding the information. S_v embeds $C\ell_{id}$ in L_o by encrypting it using Digital Watermarking Technique (DWT) and Watermarked Document (WMD) is obtained. WMD is encrypted by applying Cryptography Technique (CT) and Encrypted WMD (∂^*(WMD)) is passed to C_j. An entry is recorded to IMT by storing $C\ell_{id}$, document-ID and hash value H generated for $C\ell_{id}$ of C_j which specifies the document having document-ID as D_i has been allocated to C_j with client-ID as $C\ell_{idj}$. Simultaneously, an entry is also recorded to Data Allocation Table (DAT) that shows the set of data allocated to $C\ell_{idj}$ $\forall$ $j = \{1,2,\ldots,m\}$. If C_j leaks D_i, then detection technique is applied to find G_c when D_i is found.

3. SECURE DATA SHARING

3.1 Information Hiding

In this phase, Digital Watermarking Technique (DWT) is used to hide the information $I_m = \{C\ell_{id}, H\}$ in L_o. DLDM uses a light weight CT for encryption instead of using heavy weight CT like RSA to reduce time complexity and cost in calculating the encrypted message. An Advanced Encryption Standard (AES-128) CT is used to encrypt $C\ell_{id}$ in combination with hashing technique (HT) for authentication purpose. The input of AES-128 are secret message $C\ell_{id}$ and the secret key K of 128 bit, which produces $\partial^*(C\ell_{id})$ as an output. A hash value H of $C\ell_{id}$ is calculated using Secure Hash Algorithm SHA-3 of 512 bit HT that is used for the authenticity of the document D_i. If C_j make changes to $C\ell_{id}$, its SHA-3 code will be changed and it can be identified that tampering has been occurred with $C\ell_{id}$. Input of SHA3-512 HT are the key $H_k \in \mathscr{H}_{\mathscr{K}}$ and $C\ell_{id}$ instead of $\partial^*(C\ell_{id})$. It will distract C_j and increase the security level of DLDM and produces the hash value H of 512 bit as an output.

Generated output $I_{m'} = \{\partial^*(C\ell_{id}), H\}$ from CT and HT are embedded into L_o in D_i using DWT. Input of DWT are D_i, H, $\partial^*(C\ell_{id})$, $W_k \in \mathscr{W}_{\mathscr{K}}$ and it produces WMD as an output. Fig. 2 represents the embedding of the secret information in the original document. Note that H is embedded in the reverse order in Watermark Embedding Process (WEP) to distract C_j. Furthermore, $C\ell_{id}$ is embedded in D_i after encrypting it that provides an additional level of security to D_i. If $C\ell_{id}$ and its H value are directly embedded in D_i then C_j can alter or remove $C\ell_{id}$ and correspondingly H. Thus it provides significant security to D_i and it becomes difficult for C_j to predict the embedded $I_{m'}$ in D_i. DWT is robust to prevent the attacks and it becomes difficult for any client C_j to make changes in embedded $I_{m'}$. This technique can detect $I_{m'}$ even after its destruction or alteration by C_j. Finally, the Watermarked Document (WMD) will be generated by WEP and passed to C_j.

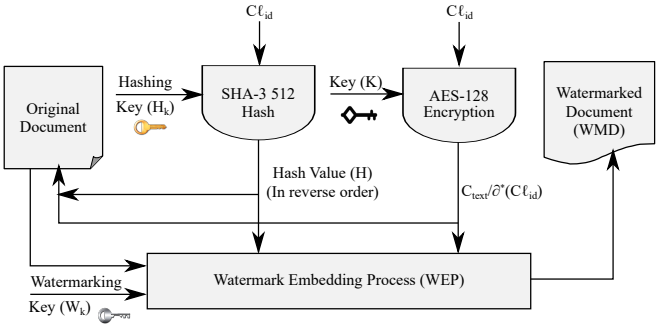


Fig. 2. Watermark embedding process using digital watermarking technique.

3.2 Securing the Data

Let $\mathscr{D}$ is the set of documents to be encrypted, $\mathscr{S}_{\mathscr{K}}$ is the set of secret keys for symmetric cryptography, $\mathscr{P}_{\mathscr{K}}$ is the set of public keys and $\mathscr{PV}_{\mathscr{K}}$ is the set of private keys for asymmetric cryptography. To provide more security to the document, generated WMD is

encrypted and then transmitted to the client C_j. The DLDM adopts a combinational CT to encrypt WMD. Although RSA is highly secure CT, but its encryption time, decryption time, memory usage, and computation cost is high. Since AES-256 is secure, faster and less expensive CT as well as utilizes less memory compared to RSA. Therefore, instead of using RSA directly to encrypt WMD, a combinational approach that utilizes both AES-256 and RSA cryptography techniques is used. We encrypt the WMD using AES-256 CT and Secret Key S_k of AES-256 will be encrypted by RSA public key P_k. The combinational approach provides the significant security to WMD as well as it is effective in term of encryption time, decryption time, memory utilization and cost.

Fig. 3 represents the sharing of D_i to C_j after encrypting it where secret key $S_k \in \mathscr{S}_{\mathscr{K}}$ is generated by AES-256 key generator and RSA key generator generates the public and private key $P_k \in \mathscr{P}_{\mathscr{K}}$, $PV_k \in \mathscr{PV}_{\mathscr{K}}$ respectively. AES-256 encrypts WMD using S_k while S_k is encrypted by RSA using P_k and encrypted WMD with secret key are transferred to client C_j where $j \in \{1,2,\ldots,m\}$.

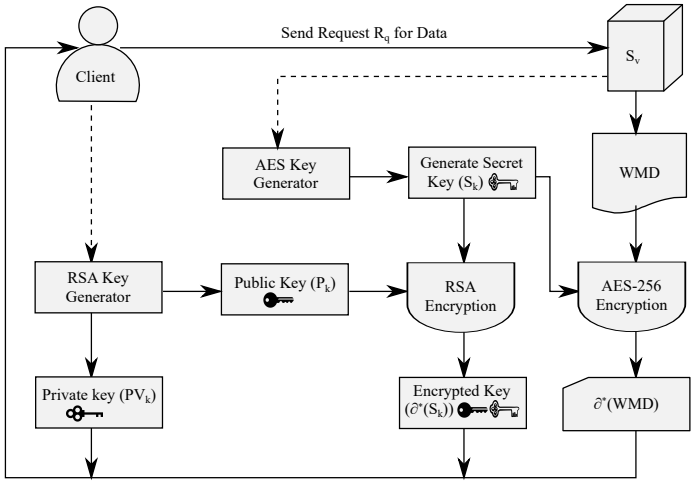


Fig. 3. Process of transferring WMD to client.

4. GUILTY CLIENT DETECTION

If C_j leaked the data at an unauthorized place then the detection mechanism is applied. The encrypted S_k is decrypted by RSA using PV_k. The resultant key S_k is used to decrypt the encrypted WMD by AES-256 and finally C_j obtains WMD. After obtaining the document, let C_j has leaked WMD to an unauthorized party. When S_v discovers the leaked data set $L_\psi \subseteq \mathbb{D}$ at an unauthorized place which is called as target t, the detection mechanism is applied to find G_c. Fig. 4 shows the complete process to detect the guilty client. $C\ell_{id}$ is obtained through watermark extraction process followed by decryption. If no tampering has been occurred with the document then G_c is identified otherwise G_c is estimated based on the calculated probability parameters.

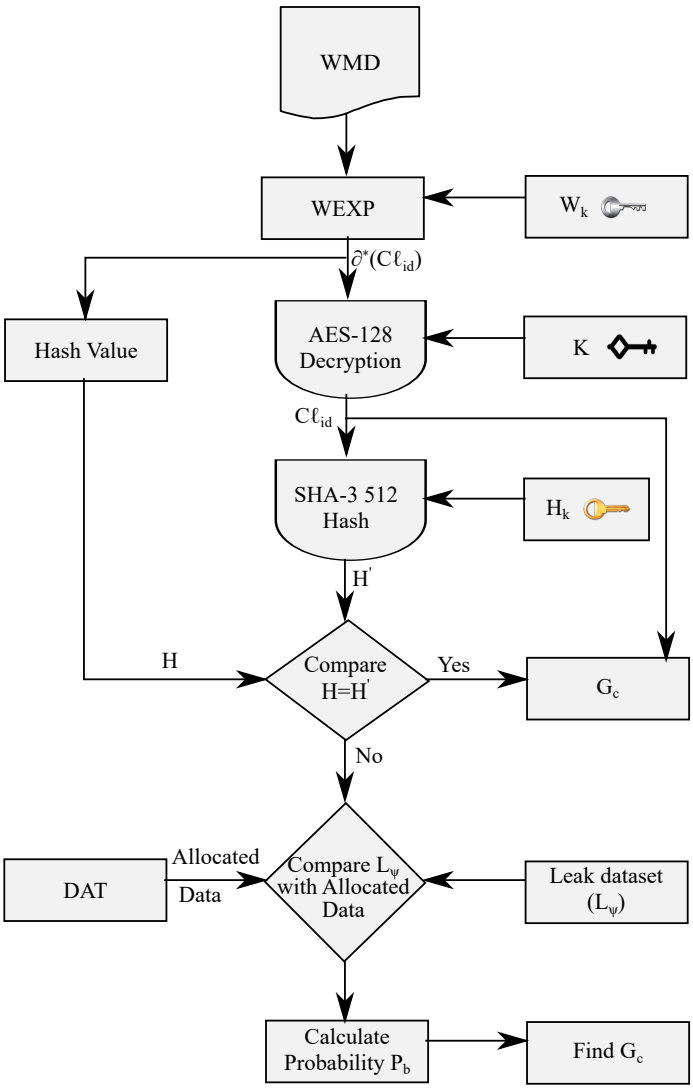


Fig. 4. A hybrid approach for detecting G_c.

4.1 Watermark Extraction

In Watermark Extraction Process (WEXP), embedded $I_{m'}$ is extracted from the discovered WMD by using DWT and H, $\partial^*(C\ell_{id})$ is obtained by extracting $I_{m'}$. Secret message $C\ell_{id}$ is obtained by decrypting C_{text} through AES-128 using key K and hash $H^{'}$ is calculated by SHA-3 512 HT. S_v verifies the obtained $C\ell_{id}$ by matching it in the stored IMT and also verifies the correctness of $C\ell_{id}$. If the calculated $H^{'}$ is same as the embedded and stored H, this mean there is no change in $C\ell_{id}$ and it is proved as a correct message. In this case, the acquired $C\ell_{id}$ is identified as G_c.

DLDM provides the significant level of security to WMD for embedding $I_{m'}$, but still there are possibilities of attacks on it and C_j can get success in modifying or removing the secret message $C\ell_{id}$ from $I_{m'}$. However, these attacks can be identified by comparing H and $H^{'}$. If $H \neq H^{'}$, it means either C_j has altered the embedded $C\ell_{id}$ or completely destroyed the watermark $I_{m'}$. In this case, G_c is identified by calculating the probability P_b which is discussed in the following subsection.

4.2 Probability Computation

Since $C_1, C_2, \ldots, C_m$ possesses the data set $Z_j \subseteq \mathbb{D}$, therefore, they can be possible candidate for being G_c. Probability P_b of C_j being G_c is calculated using Eq. (1) where $|X_{D_i}|$ denotes the number of agents having object D_i and β is the probability of guessing the data D_i.

$$P_b\left\{G_{C_j}|L_\psi\right\} = 1 - \prod_{D_i \in L_\psi \cap Z_j} \left(1 - \frac{(1-\beta)}{|X_{D_i}|}\right) \forall\ j \in \{1,2,\ldots,m\} \tag{1}$$

If C_j leaks all the data from its allocated set Z_j then the probability $P_b\left\{G_{C_j}|Z_j\right\}$ is assessed where $L_\psi = Z_j$. To maximize the possibility of identifying G_c who leaked all the allocated data, a *difference function* $\varphi^*_{(j,k)}(G_c)$ is evaluated using Eq. (2).

$$\varphi^*_{(j,k)}(G_c) = P_b\left\{G_{C_j}|Z_j\right\} - P_b\left\{G_{C_k}|Z_j\right\}\ \forall\ j,k = \{1,2,\ldots,m\} \tag{2}$$

This function results into a matrix form $\mathbb{R}^{m\times m}$ where each entry $r_{jk} \in \mathbb{R}^{m\times m} \geq 0$. If $\varphi^*_{(j,k)}(G_c) = 0$ then chances for C_j and C_k of being G_c are equal because $L_\psi \subseteq Z_j, Z_k$. In this case it becomes difficult to distinguish G_c. If $\varphi^*_{(j,k)}(G_c) > 0$ then the larger is the value of $\varphi^*_{(j,k)}(G_c)$, easier will be to identify C_j as G_c. To evaluate the performance of the proposed technique, another two parameters *average success rate* $\overline{\varphi}^*$ and *detection rate* $\min\varphi^*$ are computed using Eqs. (3) and (4) respectively. For instance, $\overline{\varphi}^* = 0.5$ means the probability $P_b\left\{G_{C_j}|Z_j\right\}$ for G_c will be 0.5 times greater than the probability of non-guilty client. $\min\varphi^*$ is the minimum of $r_{jk} \in \mathbb{R}^{m\times m}$ and it is useful in the situation where C_j leaks the data and the client C_j and C_k where $C_j \neq C_k$ have very similar probability P_b of being G_c. If value of $\min\varphi^* = 0$ then it becomes difficult to differentiate the leaker between C_j and C_k. If the value of $\min\varphi^* > 0$ then probability (P_b) of guilty client is more than the probability of non-guilty client.

$$\overline{\varphi}^* = \frac{\sum\limits_{\substack{j,k=\{1,2,\ldots,m\}\\ j\neq k}} \varphi^*_{(j,k)}(G_c)}{m(m-1)} \tag{3}$$

$$\min\varphi^* = \min_{\substack{j,k=\{1,2,\ldots,m\}\\ j\neq k}} \varphi^*_{(j,k)}(G_c) \tag{4}$$

5. PERFORMANCE EVALUATION

5.1 Experimental Set-up and Benchmark

The experiments are performed on the machines running 32 bits kernel version 6.3 with an intel ®core ™i5-2600 CPU @ 3.4 GHZ having 8 GB RAM. The cloud environment is created using 8 different machines and the prototype system is implemented in C/ C++. The crypto ++ library is used to implement the encryption and hashing schemes whereas the matlab is used for watermarking the data. We have taken Sherweb [19] which is based on Azure, Amazon, Digital Ocean statistical comparisons and cloud servers to create a virtualized atmosphere for experiments. It uses authenticated open data accessible on data.gov [20] to collect the dataset. The designed benchmark for DLDM is dealing with medical datasets as these datasets are universally alleged as most sensitive data in nature. CT scan image that represent the sensitive data is taken as a benchmark for embedding the information of client.

5.2 Data Allocation

We considered seven operations for the allocation of document D_i which is requested by C_j are: (1) Cloud DB search; (2) $C\ell_{id}$ generation; (3) $C\ell_{id}$ encryption; (4) hashing on $C\ell_{id}$; (5) watermarking; (6) document encryption; (7) RSA encryption on AES-256 keys and then computed the execution time of each phase. It can be seen from Table 1 that when the documents are provided to distinct users, the computation time increases with respect to the size and number of documents (N_D). When the documents of fixed size (20MB) are provided to single and multiple users, it is noticed that the computation time increases with a very slow rate with respect to the number of users (N_U).

Table 1. Computation time w.r.t. number of documents for different document sizes and different number of users.

N_D	Computation time (ms)							
	Size of document				Number of users (N_U)			
	40MB	60MB	80MB	100MB	1	2	4	8
20	641.2	712.6	808.7	886.6	248.1	272.1	304.1	376.5
40	1277.4	1386.8	1606.5	1767.5	491.3	498.4	530.0	609.8
60	1884.4	2097.2	2385.1	2635.2	697.5	745.3	785.2	820.9
80	2575.6	2763.6	3182.1	3497.7	964.9	997.9	1029.6	1077.8
100	3171.2	3525.9	4039.8	4404.8	1158.3	1165.2	1227.1	1326.9
120	3825.2	4140.3	4774.2	5307.2	1426.4	1447.3	1444.0	1521.7
140	4562.4	4955.9	5632.1	6160.4	1629.6	1717.2	1693.9	1818.8
160	5161.4	5613.8	6472.2	7074.3	1849.3	1867.1	1984.3	1993.9
180	5727.6	6287.5	7056.7	7768.9	2121.9	2172.4	2115.9	2170.3
200	6406.4	6922.0	8000.5	8719.2	2289.8	2352.7	2336.3	2417.9

Figs. 5 (a) and (b) represents the execution time of every phase of the proposed scheme when varieties of documents are provided to the single and distinct users respectively. It is observed from Fig. 5 (a) that the execution time of every phase is linear with

respect to the number of documents. Note that the execution time of $C\ell_{id}$ generation is increasing negligibly as compared to the other phases. From the Fig. 5 (b), it is observed that the execution time of $C\ell_{id}$ generation, $C\ell_{id}$ encryption, hashing on $C\ell_{id}$, watermarking and RSA encryption on AES-256 keys become constant. As these operations are performed single time only when the documents are provided to single user, hence the execution time for these operations are same in spite of the number of documents. In Fig. 5 (c), the computation time of every phase is calculated for the single document of various sizes. It is noted that the computation time for document for all the operations except encryption and decryption stay constant as these operations are independent of document size. In Fig. 5 (d), a single document of fixed size 20 MB is shared among number of users. In this case, the extraction and detection are not computed because the number of leaked documents are not identified. Note that all the operations except cloud DB search are linear with respect to the number of users. The execution time for DB search remains constant as searching for a document D_i will be performed single time only and the same document is shared among all the users.

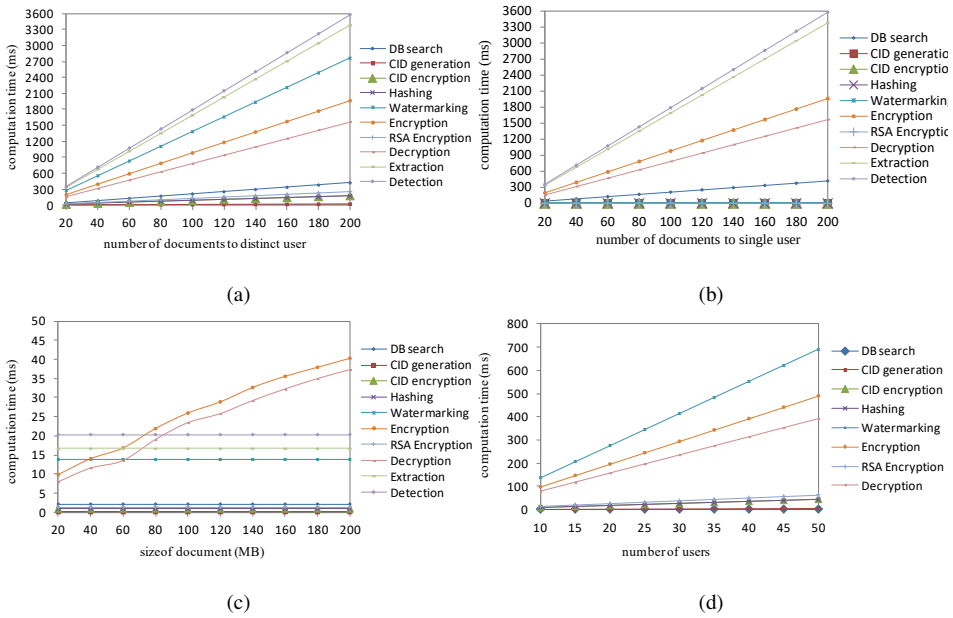
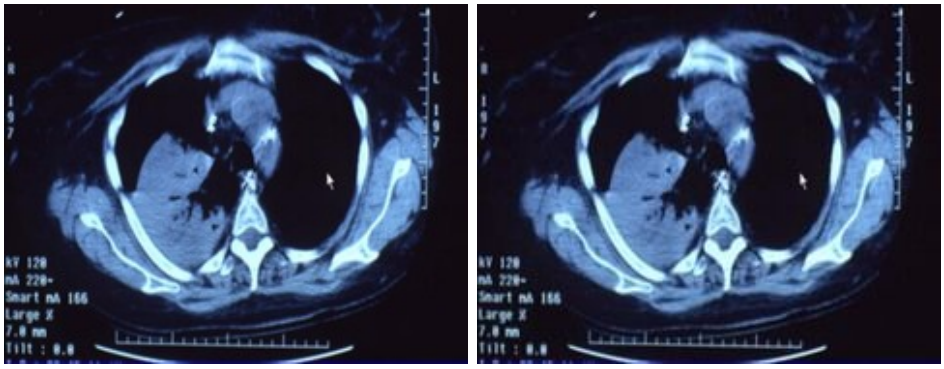


Fig. 5. Computation time for (a) different number of documents to distinct user; (b) different number of documents to single user; (c) different document sizes; (d) different number of users.

Figs. 6 (a) and (b) represent the original and the watermarked CT scan image respectively. The calculated higher $PSNR = 37.89$ describes the quality of the watermarked Image while the lower $BER = 00.28$ indicates the robustness of extracting the watermark. This is the reason that both figures appear identical and it becomes difficult for any C_j to identify the embedded $I_{m'}$. In Fig. 7 (a), the overall computation time gradually decreases when $|S_v|$ goes on increasing as the task is shared among S_v and work is performed parallelly.



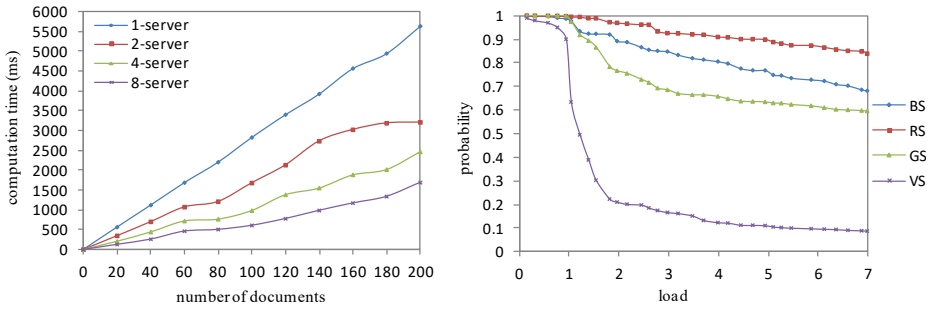
(a) (b)

Fig. 6. CT scan image (a) Original; (b) Watermarked.

5.3 Guilty Client Probability

To evaluate the parameters for G_C detection, $|\mathbb{D}| = 80$, $|\mathbb{C}| = (9-40)$, $|L_\psi| = 50$ and $\beta = 0.1$ has been considered. The performance is evaluated against the parameter load $\mathscr{L}_\mathtt{d} = \Sigma_{j=1}^{m} |Z_j| / |\mathbb{D}|$ that can be defined as the ratio of total count of data objects allocated to all the clients to the number of data objects.

In Fig. 7 (b), BS, RS, GS and VS represents the curve for $P_b\{G_{C_j}|L_\psi\}$, $P_b\{G_{C_j}|Z_j\}$, $\overline{\varphi}^*$ and $\min\varphi^*$ relatively for the various values of $\mathscr{L}_\mathtt{d}$. It is observed that average $P_b\{G_{C_j}|L_\psi\}$, average $P_b\{G_{C_j}|Z_j\}$, $\overline{\varphi}^*$ and $\min\varphi^*$ decrease correspondingly with increment in $\mathscr{L}_\mathtt{d}$. We have $P_b\{G_{C_j}|L_\psi\} = 0.723274$, $P_b\{G_{C_j}|Z_j\} = 0.864934$, $\overline{\varphi}^* = 0.610327$ and $\min\varphi^* = 0.1$ for $\mathscr{L}_\mathtt{d} = 6.125$. It is noticed that the values of all the parameters for detecting G_c is significantly high even though $\mathscr{L}_\mathtt{d}$ is very high.



(a) (b)

Fig. 7. (a) Comparison of task distribution among different servers; (b) Evaluation of probability for G_c detection.

6. CONCLUSION AND FUTURE DIRECTIONS

To address the data leakage problem, DLDM is presented that assesses the likelihood of a user for being guilty entity and detects the malicious user who has leaked the confidential information at some unauthorized place through incidental exposure or intentional sabotage. The proposed approach provides the security to protect the confidential information. To detect the guilty entity, the model considered a vigorous combination of cryptography, hashing, watermarking and the probability techniques. DLDM considered the documents of various types and sizes to evaluate the performance. The results show that the proposed model is cost effective in terms of computation time. This work can be further extended for more complex threat model.

REFERENCES

1. J. Wei, W. Liu, and X. Hu, "Secure data sharing in cloud computing using revocablestorage identity-based encryption," *IEEE Transactions on Cloud Computing*, Vol. 6, 2018, pp. 1136-1148.
2. W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, Vol. 14, 2019, pp. 331-346.
3. Privacy rights clearinghouse (prc), https://www.privacyrights.org/data-breaches.
4. Annual cost of a data breach study: Global overview, https://www.ibm.com/downloads/cas/861MNWN2.
5. I. Gupta and A. K. Singh, "Dynamic threshold based information leaker identification scheme," *Information Processing Letters*, Vol. 147, 2019, pp. 69-73.
6. N. Kumar, V. Katta, H. Mishra, and H. Garg, "Detection of data leakage in cloud computing environment," in *Proceedings of International Conference on Computational Intelligence and Communication Networks*, 2014, pp. 803-807.
7. Y. Kao, K. Huang, H. Gu, and S. Yuan, "ucloud: a user-centric key management scheme for cloud data protection," *IET Information Security*, Vol. 7, 2013, pp. 144-154.
8. A. Al-Haj, G. Abandah, and N. Hussein, "Crypto-based algorithms for secured medical image transmission," *IET Information Security*, Vol. 9, 2015, pp. 365-373.
9. S.-F. Tan and A. Samsudin, "Ciphertext policy-attribute based homomorphic encryption (cp-abher-lwe) scheme: A fine-grained access control on outsourced cloud data computation," *Journal of Information Science and Engineering*, Vol. 33, 2017, pp. 675-694.
10. P. Pandiaraja, P. Vijayakumar, V. Vijayakumar, and R. Seshadhri, "Computation efficient attribute based broadcast group key management for secure document access in public cloud," *Journal of Information Science and Engineering*, Vol. 33, 2017, pp. 695-712.
11. S. Bishop, H. Okhravi, S. Rahimi, and Y.-C. Lee, "Covert channel resistant information leakage protection using a multi-agent architecture," *IET Information Security*, Vol. 4, 2010, pp. 233-247.

12. M. Backes, N. Grimm, and A. Kate, "Data lineage in malicious environments," *IEEE Transactions on Dependable and Secure Computing*, Vol. 13, 2016, pp. 178-191.
13. I. Gupta, N. Singh, and A. Singh, "Layer-based privacy and security architecture for cloud data sharing," *Journal of Communications Software and Systems*, Vol. 15, 2019.
14. I. Gupta and A. K. Singh, "A probabilistic approach for guilty agent detection using bigraph after distribution of sample data," *Proceedings of the 7th International Conference on Smart Computing and Communications*, 2017.
15. P. Papadimitriou and H. Garcia-Molina, "Data leakage detection," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, 2011, pp. 51-63.
16. A. Kumar, A. Goyal, A. Kumar, N. K. Chaudhary, and S. S. Kamath, "Comparative evaluation of algorithms for effective data leakage detection," in *Proceedings of IEEE Conference on Information Communication Technologies*, 2013, pp. 177-182.
17. S. Sodagudi and R. R. Kurra, "An approach to identify data leakage in secure communication," in *Proceedings of the 2nd International Conference on Intelligent Computing and Applications*, 2017, pp. 31-43.
18. I. Gupta and A. K. Singh, "A probabilistic approach for guilty agent detection using bigraph after distribution of sample data," *Procedia Computer Science*, Vol. 125, 2018, pp. 662-668.
19. Sherweb, https://www.sherweb.com/en-eu/.
20. Open data, https://data.gov.in/catalogs.

Ishu Gupta received the BCA, M.Sc and MCA (Gold Medalist) degrees in Computer Science from Kurukshetra University, India. Currently, she is working as a Ph.D. student in the Department of Computer Applications, National Institute of Technology, Kurukshetra, India. She is awarded with Senior Research Fellowship by the University Grant Commission, India. Her recent research interests include the areas of cloud computing and information security.

Ashutosh Kumar Singh is working as a Professor and Head in the Department of Computer Applications, National Institute of Technology Kurukshetra, India. He received his Ph.D. in Electronics Engineering from Indian Institute of Technology, BHU, India and Post Doc from University of Bristol, UK. His research area includes Cloud Computing, Machine Learning, Security, Big Data, Verification, Synthesis, Design and Testing of Digital Circuits. He has published more than 200 research papers in different journals, conferences, book chapters, and news magazines.