ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Enabling Cost-Effective Privacy Preserving Of Intermediate Sensitive Data Sets in Cloud Using Proxy Re-Encryption Technique

A.Jeeva[1]

M.E. Student, Department of CSE, Sri Eshwar College of Engineering, Anna University - Chennai, India[1]

ABSTRACT: Propose an upper bound privacy leakage constraint based approach to find which dataset is need to encrypted. Because encrypting all datasets is very time consuming. Encrypted dataset is outsourced in different cloud providers. The datasets are divided into several parts and stored in different cloud storage. So the privacy preserving cost can be saved. To preserve privacy, the user will encrypt their data and re-encrypted form of data greatly impedes the utilization due to its randomness. So the data stored in cloud only on encrypted form. The data can be accessible only to users with the correct keys. The proxy re-encryption techniques can prevent against the errors and attacks.

KEYWORDS: Cloud Computing, Data Storage Privacy, Privacy Preserving, Intermediate Dataset, Proxy Re-encryption

I. INTRODUCTION

Cloud computing is offering IT services and data storage services over a network. Participants in the business chain of cloud computing can benefit from this novel model. Cloud customers can save huge capital investment of IT infrastructure, and concentrate on their own core business. Therefore, many companies or organizations have been migrating or building their business into cloud. For example, the email service is probably the most popular one. Cloud computing is a concept that treats the resources on the Internet as a unified entity, a cloud. Users just use services without being concerned about how computation is done and storage is managed. However, numerous potential customers are still hesitant to take advantage of cloud due to security and privacy concerns. The privacy concerns caused by retaining intermediate datasets in cloud are important but they are paid little attention. Storage and computation services in cloud are equivalent from an economical perspective because they are charged in proportion to their usage. This paper focuses on designing a cloud storage system for robustness, confidentiality, and functionality. A cloud storage system is considered as a large scale distributed storage system that consists of many independent storage servers. Thus, cloud users can store valuable intermediate datasets selectively when processing original datasets in data-intensive applications like medical diagnosis, in order to curtail the overall expenses by avoiding frequent re-computation to obtain these datasets. Such scenarios are quite common because data users often re-analyze results, conduct new analysis on intermediate datasets, or share some intermediate results with others for collaboration. It is very robust because the message can be retrieved as long as one storage server survives. Another way is to encode a message of k symbols into a codeword of n symbols by erasure coding. To store a message, each of its codeword symbols is stored in a different storage server. A storage server failure corresponds to an erasure error of the codeword symbol. As long as the number of failure servers is under the tolerance threshold of the erasure code, the message can be recovered from the codeword symbols stored in the available storage servers by the decoding process. datasets stored in cloud mainly include encryption and anonymization.

On one hand, encrypting all datasets, a straightforward and effective approach, is widely adopted in current research. However, processing on encrypted datasets efficiently is quite a challenging task, because most existing applications only run on unencrypted datasets. Although recent progress has been made in homomorphic encryption which theoretically allows performing computation on encrypted datasets, applying current algorithms are rather expensive due to their inefficiency. On the other hand, partial information of datasets, e.g., aggregate information, is required to expose to data users in most cloud applications like data mining and analytics. In such cases, datasets are anonymized rather than encrypted to ensure both data utility and privacy preserving. Current privacy preserving techniques like generalization can withstand most privacy attacks on one single dataset, while preserving privacy for multiple datasets is still a challenging problem. Thus, for preserving privacy of multiple datasets, it is promising to anonymize all datasets first and then encrypt them before storing or sharing them in cloud. Usually, the volume of intermediate datasets is huge. Hence, we argue that encrypting all intermediate datasets will lead to high overhead and low efficiency when they are frequently accessed or processed. As such, we propose to encrypt part of intermediate datasets rather than all for reducing privacy-preserving cost.

II. OBJECTIVE OF THE WORK

The objective of this project is to encrypt the data sets and stored in different cloud environment using proxy reencryption technique. Proxy reencryption technique to improve the Project tasks is as follows:

- Analyse the dataset which is need to encrypt and others are not encrypted. This analyse make sure the data need to encrypted.

- Review the research on privacy protection and consider the economical aspect of privacy preserving, adhering to the pay-as-you-go feature of cloud computing.

- Once identify the data to be encrypted, we must choose how many keys to use for encryption, and the granularity of encryption.

- Encrypt all such data using a single key, and share the key with all users of the service. Unfortunately, this has the problem that a malicious or compromised cloud could obtain access to the encryption key, e.g. by posing as a legitimate user, or by compromising or coluding with an existing user.
- Confidentiality of the entire dataset would be compromised. In the other extreme, we could encrypt each data object with a different key. This increases robustness to key compromise, but drastically increases key management complexity.

- Our goal is to automatically infer the right granularity for data encryption that provides the best tradeoff between robustness and management complexity.

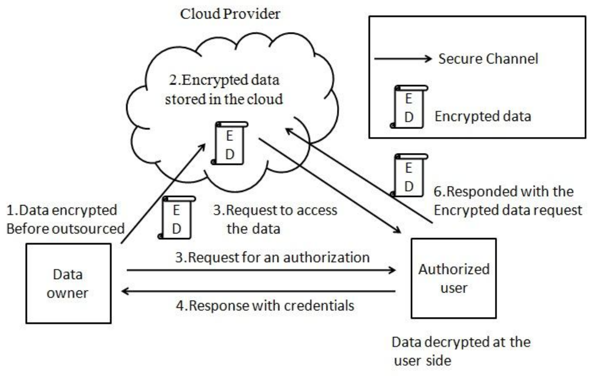
Cloud Provider
2.Encrypted data stored in the cloud
E D
Secure Channel
E D Encrypted data
E D
6.Responded with the Encrypted data request
1.Data encrypted Before outsourced
E D
3.Request to access the data
3.Request for an authorization
Data owner
4.Response with credentials
Authorized user
Data decrypted at the user side

Fig 2.1 Privacy Preserving in Cloud

III. RELATED WORK

Briefly review distributed storage systems, proxy re-encryption schemes and decentralized erasure code.

3.1 Distributed Storage Systems

The Network-Attached Storage (NAS) and the Network File System (NFS) provide storage devices over the network such that a user can access the storage devices via network connection. A decentralized architecture for storage systems offers good scalability, because a storage server can join or leave without control of a central authority. To provide robustness against server failures, a simple method is to make replicas of each message and store them in different servers. One way to reduce the expansion rate is to use erasure codes to encode messages. A message is encoded as a codeword, which is a vector of symbols, and each storage server stores a codeword symbol. A storage server failure is modeled as an erasure error of the stored codeword symbol. To store a message of k blocks, each storage server linearly combines the blocks with randomly chosen coefficients and stores the codeword symbol and coefficients. To retrieve the message, a user queries k storage servers for the stored codeword symbols and coefficients and solves the linear system. The system has light data confidentiality because an attacker can compromise k storage servers to get the message.

3.2 Proxy Re-Encryption Scheme

In a proxy re-encryption scheme, a proxy server can transfer a cipher text under a private key A to a new one under another public key B. The server does not know the plaintext during transformation. The data is first encrypted with a symmetric data encryption key and then stored in the cloud storage server. The cloud storage server uses a re-encryption algorithm to transfer the encrypted DEK into the format that can be decrypted by the recipient's private key. The recipient then can download the encrypted data from the cloud and use the DEK for decryption. A re-encryption key is generated from the data owner's private key and a recipient's public key. A data owner may share different files with different recipient groups. Therefore, a recipient cannot read data for a group it does not belong to. The cloud, on the other hand, acts as an intermediate proxy. It cannot read the data as it cannot get DEKs. Thus the system has data confidentiality and supports the data forwarding function.

3.3 Decentralized Erasure Code

A decentralized erasure code is a random linear code with a sparse generator matrix. The generator matrix G constructed by an encoder is as follows: First, for each row, the encoder randomly marks an entry as 1 and repeats this process for an ln k/k times with replacement. Second, the encoder randomly sets a value from IF for each marked entry. This finishes the encoding process. A decoding is successful if and only if $k \times k$ submatrix formed by the k-chosen columns is invertible. Thus, the probability of a success decoding is the probability of the chosen sub matrix being invertible. The owner randomly selects v servers with replacement and sends a copy of Mi to each of them. Each server randomly selects a coefficient for each received cipher text and performs a linear combination of all received cipher texts. Those coefficients chosen by a server form a column of the matrix and the result of the linear combination is a codeword element. Each server can perform the computation independently. This makes the code decentralized.

IV. THREAT MODEL

We consider data confidentiality for both data storage and data forwarding. In this threat model, an attacker wants to break data confidentiality of a target user. To do so, the attacker colludes with all storage servers, non-target users, and up to (t-1) key servers. The attacker analyzes stored messages in storage servers, the secret keys of non-target users, and the shared keys stored in key servers. Note that the storage servers store all re-encryption keys provided by users. The attacker may try to generate a new re-encryption key from stored re encryption keys. We formally model this attack by the standard chosen plaintext attack1 of the proxy re-encryption scheme in a threshold version. A cloud storage system modeled in the above is secure if no probabilistic polynomial time attacker wins the game with a non negligible advantage. A secure cloud storage system implies that an unauthorized user or server cannot get the content of stored messages, and a storage server cannot generate re-encryption keys by himself. If a storage server can generate a re-encryption key from the target user to another user B, the attacker can win the security game by re-encrypting the cipher text to B and decrypting the re-encrypted cipher text using the secret key SKB. Therefore, this model addresses the security of data storage and data

V. EXPERIMENT EVALUATION

5.1 Experiment Environment

U-Cloud is a cloud computing environment at University of Technology Sydney (UTS). The system overview of UCloud is depicted in The computing facilities of this system are located among several labs at UTS. On top of hardware and Linux operating system, we install KVM virtualization software [30] which virtualizes the infrastructure and provides unified computing and storage resources. To create virtualized data centers, we install Open Stack open source cloud environment for global management, resource scheduling and interaction with users. Further, Hadoop is installed based on the cloud built via Open Stack to facilitate massive data processing. Our experiments are conducted in this cloud environment. To demonstrate the effectiveness and scalability of our approach, we run H_PPCR and ALL_ENC on real-world datasets and extensive datasets. We first them on real-world datasets with $\varepsilon\varepsilon dd$ varying in [0.05, 0.9], then on extensive intermediate datasets with the number ranging in [50, 1000] under certain $\varepsilon\varepsilon dd$. For each group of experiments, we repeat H_PPCR and ALL_ENC 50 times. The mean of cost in each experiment group are regarded as the representative. The margin of error with 95% confidence is also measured and shown in the results. We first conduct our experiments on the Adult dataset which a commonly-used dataset in the privacy research community. Intermediate datasets are generated from the original dataset, and anonymized by the algorithm proposed in. Further, we extend experiments to intermediate data-sets of large amounts. SITs are generated via a random spanning tree algorithm. The values of data size and usage frequencies are randomly produced in the interval [10, 100] according to the uniform distribution.

5.2. Experiment Results and Analysis

The experimental result on real-world datasets is depicted from which we can see that $CCHHEEUU$ is much lower than $CCSSPPPP$ with different privacy leakage degree. Even the smallest cost saving of $CCHHEEUU$ over $CCSSPPPP$ at the left side is more than 40%.Further, we can see that the difference $CCSSSSVV$ between $CCSSPPPP$ and $CCHHEEUU$ increases when the privacy leakage degree increases. This is because looser privacy leakage restraints imply more datasets can remain unencrypted. With Fig.5 where we reason about the difference between $CCHHEEUU$ and $CCSSPPPP$ with different privacy leakage degree, illustrates how the difference changes with different numbers of extensive datasets while $\varepsilon\varepsilon dd$ is certain. In most real-world cases, data owners would like the data privacy leakage to be much low. Thus, we select four low privacy leakage degrees of 0.01, 0.05, 0.1 and 0.2 to conduct our experiments. The selection of these specific values is rather random and does not affect our analysis because what we want to see is the trend of $CCHHEEUU$ against $CCSSPPPP$. Similarly, we set the number of $\varepsilon\varepsilon dd$ values as four.

VI. CONCLUSTION

Proposed an approach that identifies which part of intermediate datasets needs to be encrypted while the rest does not, in order to save the privacy preserving cost. A tree structure has been modeled from the generation relationships of intermediate datasets to analyze privacy propagation among datasets. We have modeled the problem of saving privacy-preserving cost as a constrained optimization problem which is addressed by decomposing the privacy leakage constraints. A practical heuristic algorithm has been designed accordingly. Evaluation results on real-world datasets and larger extensive datasets have demonstrated the cost of preserving privacy in cloud can be reduced significantly with our approach over existing ones where all datasets are encrypted. In accordance with various data and computation intensive applications on cloud, intermediate dataset management is becoming an important research area. Privacy preserving for intermediate datasets is one of important yet challenging research issues, and needs intensive investigation. With the contributions of this paper, we are planning to further investigate privacy-aware efficient scheduling of intermediate datasets in cloud by taking privacy preserving as a metric together with other metrics such as storage and computation. Optimized balanced scheduling strategies are expected to be developed towards overall highly efficient privacy aware dataset scheduling.

REFERENCES

[1] I. Roy, S.T.V. Setty, A. Kilzer, V. Shmatikov and E. Witchel, “Airavat: Security and Privacy for Mapreduce,” Proc. 7th USENIX Conf. Networked Systems Design and Implementation (NSDI'10), pp. 20-20, 2010.

[2] K.P.N. Puttaswamy, C. Kruegel and B.Y. Zhao, “Silverline: Toward Data Confidentiality in Storage-Intensive Cloud Applications,” Proc. 2nd ACM Symp. Cloud Computing (SoCC'11), 2011.

[3] K. Zhang, X. Zhou, Y. Chen, X. Wang and Y. Ruan, “Sedic: Privacy-Aware Data Intensive Computing on Hybrid Clouds,” Proc. 18th ACM Conf. Computer and Communications Security (CCS'11), pp. 515-526, 2011.

[4] V. Ciriani, S.D.C.D. Vimercati, S. Foresti, S. Jajodia, S. Paraboschi and P. Samarati, “Combining Fragmentation and Encryption to Protect Privacy in Data Storage,” ACM Trans. Information and System Security, vol. 13, no. 3, pp. 1-33, 2010.

[5] S.B. Davidson, S. Khanna, T. Milo, D. Panigrahi and S. Roy, “Provenance Views for Module Privacy,” Proc. 30th ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems (PODS'11), pp. 175-186, 2011.

[6] S.B. Davidson, S. Khanna, S. Roy, J. Stoyanovich, V. Tannen and Y. Chen, “On Provenance and Privacy,” Proc. 14th Int'l Conf. Database Theory, pp. 3-10, 2011.

[7] S.B. Davidson, S. Khanna, V. Tannen, S. Roy, Y. Chen, T. Milo and J. Stoyanovich, “Enabling Privacy in Provenance-Aware Workflow Systems,” Proc. 5th Biennial Conf. Innovative Data Systems Research (CIDR'11), pp. 215-218, 2011. [22] P. Samarati, “Protecting Respondents' Identities in Microdata Release,” IEEE Trans. Knowl. Data Eng., vol. 13, no. 6, pp. 1010- 1027, 2001.