Fake Objects Based Intruder Detection System In Cloud Storage

K V Rajani[1] , V.Sangeetha[2,] K.J Archana[3]
[1]CSE Department, Swami Vivekananda Institute of Technology SVIT), Secunderbaad-26,

Abstract: *Demanding of cloud conditions motivated many organizations to outsource their sensitive data to the cloud storage. Once outsourcing the data to the cloud the owner may share his data to set of authorized users, some of the data may be found in an authorized place now the data owner has to find the guilty user who leaked the data. Now the data owner also identifies enough evidence that the user leaked. So in this paper we develop a model for evaluate the guilt user. According to existing Perturbation and water mark techniques are used to detect the guilt user but these model will not efficient to identify the guilt user in cloud environment. In order to find guilt agent we present an algorithm for identifying the guilt user and this algorithm will improves the chances of detection of leakage user. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can sometimes be destroyed if the user is malicious*.

Keywords: watermark, Sensitive Data, Fake Objects, Data Allocation Strategies

INTRODUCTION:

In the cloud storage service, sometimes sensitive data must be outsourced to the cloud. For example, a hospital may outsource patient health records to cloud and it will used be doctors who will devise new treatments. Similarly, a business may have partnership with other companies that require sharing customer data through cloud. Another enterprise may outsource its data processing, so data must be given to various other companies. Our goal is to detect when the owner sensitive data has been leaked by users, and if possible to identify the user that leaked the data.

We believe cloud where the original sensitive data cannot be perturbed. Perturbation is a very useful technique where the data is customized and prepared "less sensitive" before being outsourced. However, in some cases it is important not to alter the original owner data. For example, if an outsourcer is doing our payroll, he must have the exact salary and customer identification numbers. If medical researchers will be treating patients (as opposed to simply computing statistics), they may need accurate data for the patients. So in perturbation technique the data will modify, perturbation is useful when outsourcing data is small if it is large amount of data it may will affect to original data.

Usually, in cloud data leakage recognition is handled by watermarking, e.g., a unique code is embedded in each shared copy. If that copy is later discovered in the hands of unauthorized users, the leaker can be identified. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can sometimes be destroyed if the data recipient is malicious.

PROBLEM DEFINITION:

In cloud environment the data owner will outsource his data to the cloud and after uploading the data can be shared to multiple authorized users sometimes may be data find in unauthorized user now the data owner has to find guilt user among all the users.

Suppose data owner have a set of documents D = {d1,d2,……,dm } where m is the number of document. First the owner wants to upload his documents to the cloud and later some of the documents he may want share with set of users U1, U2…, Un but the same document one of the authorized user shared with un authorized users.

The user Ui receives a subset of documents dm which belongs to D, determined either by a sample request or an explicit request.

Sample Request Ri = SAMPLE (D,mi) : Any subset of mi records from D can be given to Ui . Explicit Request Ri = EXPLICIT (D, condition) : user Ui receives all the D document that satisfy condition .

GUILTY AGENTS:

The documents in D could be of any type and size, e.g., they could be tuples in a relation, or relations in a database. After giving documents to users, the owner discovers that a set S of D has leaked. This means that some third party called the target has been caught in possession of S. For example, this target may be displaying S on its web site, or perhaps as part of a legal discovery process, the target turned over S to the owner. Since the users U1,U2 ,… ,Un, have some of the data, it is reasonable to suspect them leaking the data. However, the owner can argue that they are innocent, and that the S data was obtained by the target through other means.

Data Allocation Problem

The main focus of our paper is the data allocation problem: how can the user "intelligently" give data to agents in order to improve the chances of detecting a guilty agent? As illustrated in Figure 1, there are four instances of this problem we address, depending on the type of data requests made by agents and whether "fake objects" are allowed.

The two types of requests we handle were defined are sample and explicit. Fake objects are objects generated by the user that are not in set T . The objects are designed to look like real objects, and are distributed to agents together with the T objects.

As shown in Figure 1, we represent our four problem instances with the names EF , EF , SF and SF , where E stands for explicit requests, S for sample requests, F for the use of fake objects, and F for the case where fake objects are not allowed.

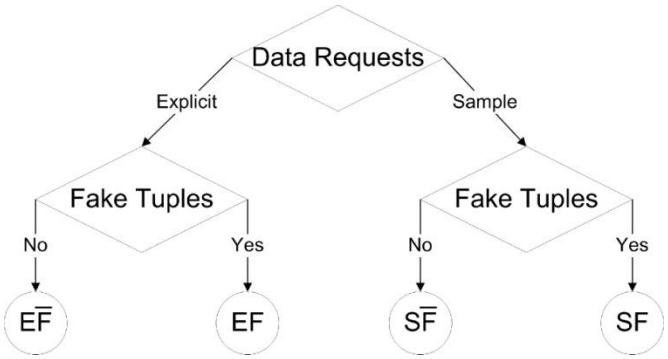

Figure 1: Leakage Problem Instances

Note that for simplicity we are assuming that in the E problem instances, all agents make explicit requests, while in the S instances, all users make sample requests. Our results can be extended to handle mixed cases, with some explicit and some sample requests. We provide here a small ex-ample to illustrate how mixed requests can be handled, but then do not elaborate further. Assume that we have two users with requests R1 = EXPLICIT(T; cond1) and R2 = SAMPLE(T 0; 1) where T 0 = EXPLICIT(T; cond2). Further, say cond1 is "state=CA" (objects have a state field). If agent U2 has the same condition cond2 = cond1, we can create an equivalent problem with sample data re-quests on set T 0. That is, our problem will be how to distribute the CA objects to two users, with R1 = SAMPLE(T 0; jT 0j) and R2 = SAMPLE(T 0; 1). If instead U2 uses condition "state=NY," we can solve two different problems for sets T 0 and T T 0. In each problem we will have only one agent. Finally,if the

conditions partially overlap, R1 \ T 0 =6 ;, but R1 =6 T 0 we can solve three different problems for sets R1 T 0, R1 \ T 0 and T 0 R1.

SYSTEM IMPLEMENTATION:

After understanding the problem statement we propose data sharing strategies advance the likelihood of identifying leakages in cloud environment. In existing mostly they adopted on data alteration but now these methods do not rely on alterations of the outsourced data. In some cases we can also inject realistic but fake data records to further improve our chances of detecting leakage and identifying the guilty party. We also present algorithm for sharing document to user. Our goal is to detect when the owner sensitive data has been leaked by users, and if possible to identify the user that leaked the data. Perturbation is a very useful technique where the data is modified and made 'less sensitive´ before being handed to users. We develop unobtrusive techniques for detecting leakage of a set of documents or records. In this section we develop a model for assessing the 'guilt´ of users. We also present algorithms for distributing documents to users, so in this process while sharing the document with authorized users the owner will add object to the document if the user will share same document with others the object count will increase so the owner can find guilt user based on object count. Today the advancement in technology made the watermarking system a simple technique of data authorization. There is various software which can remove the watermark from the data and makes the data as original.

The implementation stage involves careful planning, investigation of the existing system and it's constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods. Below are the some entities who are involved in this process.

(1) Data owner

(2) Authorized user

(3) Data Leakage & Detection

Data owner: owner can outsource his sensitive data to the cloud whenever required he can share to group of authorized users.

Authorized users: the user can download data from cloud and they can also share to authorized users or they can share to unauthorized users.

Data Leakage and Detection:

Whenever owner data is discovered other than authorized users then the owner assumes one of the authorized user committed fraud. Now for detecting the guilt user the owner will user fake object count.

Conclusion:

In cloud environment may sensitive data outsourced to cloud and it may be accessed by different users that may unknowingly or maliciously leak it. And even if we had to hand over sensitive data, in a cloud we could watermark each object so that we could trace its origins with absolute certainty. However, in many cases we must indeed work with users that may not be 100% trusted, and we may not be certain if a leaked data came from a user or from some other users. In spite of these difficulties, we have shown it is possible to assess the likelihood that and user is responsible for a leak, based on the overlap of his data with the leaked data and the data of other users, and based on the probability that objects can be "guessed" by other means. Our model is relatively simple, but we believe it captures the essential tradeoffs.

References:

[1] M. Alawneh and I.M. Abbadi, "Preventing information leakage between collaborating organisations", In Proceedings of the 10th international Conference on Electronic Commerce, vol. 342, pp. 1-10, 2008.

[2] M. Alawneh and I.M. Abbadi, "Preventing Insider Information Leakage for Enterprises", The Second International Conference on Emerging Security Information, Systems and Technologies, pp. 99-106, 2008.

[3] S. Cabuk, "Network Covert Channels: Design, Analysis, Detection, and Elimination", PhD Thesis, Purdue University, 2006. [4] S. Cabuk , C. Brodley, and C. Shields, "IP covert timing channels: Design

and detection", In proceedings of the 2004 ACM Conference on Computer and Communications Security, pp. 178-187, 2004.

[4] P. Buneman and W.-C. Tan. Provenance in databases. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 1171–1173, New York, NY, USA, 2007. ACM.

[5] Y. Cui and J. Widom. Lineage tracing for general data warehouse transformations. In The VLDB Journal, pages 471–480, 2001.

[6] S. Czerwinski, R. Fromm, and T. Hodes. Digital music distribution and audio watermarking.

[7] F. Guo, J. Wang, Z. Zhang, X. Ye, and D. Li. Information Security Applications, pages 138–149. Springer, Berlin / Heidelberg, 2006. An Improved Algorithm to Watermark Numeric Relational Data.

[8] F. Hartung and B. Girod. Watermarking of uncompressed and compressed video. Signal Processing, 66(3):283–301, 1998.

[9] S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. ACM Trans. Database Syst., 26(2):214–260, 2001.

[10] Y. Li, V. Swarup, and S. Jajodia. Fingerprinting relational databases: Schemes and specialties. IEEE Transactions on Dependable and Secure Computing, 02(1):34–45, 2005.