International Journal of Computer Applications in Engineering Sciences
[VOL I, ISSUE II, JUNE 2011] [ISSN: 2231-4946]

Development of Data leakage Detection Using Data Allocation Strategies

Rudragouda G Patil

Dept of CSE,
The Oxford College of Engg, Bangalore.

patilrudrag@gmail.com

***Abstract*-A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). If the data distributed to third parties is found in a public/private domain then finding the guilty party is a nontrivial task to distributor. Traditionally, this leakage of data is handled by water marking technique which requires modification of data. If the watermarked copy is found at some unauthorized site then distributor can claim his ownership. To overcome the disadvantages of using watermark [2], data allocation strategies are used to improve the probability of identifying guilty third parties. In this project, we implement and analyze a guilt model that detects the agents using allocation strategies without modifying the original data. The guilty agent is one who leaks a portion of distributed data. The idea is to distribute the data intelligently to agents based on sample data request and explicit data request in order to improve the chance of detecting the guilty agents. The algorithms implemented using fake objects will improve the distributor chance of detecting guilty agents. It is observed that by minimizing the sum objective the chance of detecting guilty agents will increase. We also developed a framework for generating fake objects.**

Keywords - sensitive data; fake objects; data allocation strategies;

I. INTRODUCTION

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. For example, a hospital may give patient records to researchers who will devise new treatments. Similarly, a company may have partnerships with other companies that require sharing customer data. We call owner of the data, the distributor and the supposedly trusted third parties the agents. The goal of project is to detect when the distributor's sensitive data has been leaked by agents, and show the probability for identifying the agent that leaked the data.

We study unobtrusive techniques for detecting leakage of a set of objects or records. Specifically, we study the following scenario: After giving a set of objects to agents, the distributor discovers some of those same objects in an unauthorized place. (For example, the data may be found on a web site, or may be obtained through a legal discovery process.) At this point the distributor can assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means. We develop a model for assessing the "guilt" of agents. We also present algorithms for distributing objects to agents, in a way that improves our chances of identifying a leakier. Finally, we also consider the option of adding "fake" objects to the distributed set.

II. PROBLEM DEFINITION

Suppose a distributor owns a set $T = \{ t_1, t_m\}$ of valuable data objects. The distributor wants to share some of the objects with a set of agents $U_1, U_2, \ldots, U_n$ but does wish the objects be leaked to other third parties. An agent U_i receives a subset of objects R_i which belongs to *T*, determined either by a sample request or an explicit request,

Sample Request R_i = SAMPLE (*T*, m_i) : Any subset of m_i records from *T* can be given to U_i.

Explicit Request R_i= EXPLICIT (*T*, $cond_i$) : Agent U_i receives all the *T* objects that satisfy $cond_i$.

The objects in *T* could be of any type and size, e.g., they could be tuples in a relation, or relations in a database. After giving objects to agents, the distributor discovers that a set *S* of *T* 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 distributor. Since the agents $U_1, U_2 \ldots U_n$, have some of the data, it is reasonable to suspect them leaking the data. However, the agents can argue that they are innocent, and that the *S* data was obtained by the target through other means.

A. *Agent Guilt Model*

Suppose an agent U_i is guilty if it contributes one

or more objects to the target. The event that agent U_i is guilty for a given leaked set *S* is denoted by G_i | *S*. The next step is to estimate Pr $\{G_i | S\}$, i.e., the probability that agent G_i is guilty given evidence *S*.

To compute the Pr $\{G_i | S\}$, estimate the probability that values in *S* can be "guessed" by the target. For instance, say some of the objects in *t* are emails of individuals. Conduct an experiment and ask a person to find the email of say 100 individuals, the person may only discover say 20, leading to an estimate of 0.2. Call this estimate as p_t, the probability that object *t* can be guessed by the target.

The two assumptions regarding the relationship among the various leakage events.

Assumption 1: For all *t*, $t \in S$ such that $t \neq \grave{t}$ the provenance of *t* is independent of the provenance of $\grave{t}$.

The term provenance in this assumption statement refers to the source of a value *t* that appears in the leaked set. The source can be any of the agents who have t in their sets or the target itself.

Assumption 2: An object $t \in S$ can only be obtained by the target in one of two ways.

- A single agent U_i leaked *t* from its own R_i set, or
- The target guessed (or obtained through other means) *t* without the help of any of the *n* agents.

To find the probability that an agent U_i is guilty given a set *S*, consider the target guessed t_1 with probability *p* and that agent leaks t_1 to *S* with the probability 1-*p*. First compute the probability that he leaks a single object *t* to *S*. To compute this, define the set of agents $V_t = \{U_i | t \in R_i\}$ that have *t* in their data sets. Then using Assumption 2 and known probability *p*, we have

Pr {some agent leaked *t* to *S*} = 1- *p* ……1.1

Assuming that all agents that belong to V_t can leak *t* to *S* with equal probability and using Assumption 2 obtain,

$$\Pr\{U_i \text{ leaked } t \text{ to } S\} = \begin{cases} \frac{1-p}{|V_t|} & if\ U_i \in V_t \\ 0, & otherwise \end{cases} \quad \text{.....1.2}$$

Given that agent U_i is guilty if he leaks at least one value to *S*, with Assumption 1 and Equation 1.2 compute the probability Pr $\{G_i | S\}$, agent U_i is guilty,

$$\Pr\{G_i | S\} = \prod_{t \in S \cap R_i} \left(1 - \frac{1-p}{|V_t|}\right) \text{ …….....1.3}$$

B. Data Allocation Problem

The distributor "intelligently" gives data to agents in order to improve the chances of detecting a guilty agent. There are four instances of this problem, depending on the type of data requests made by agents and whether "fake objects" [4] are allowed. Agent makes two types of requests, called sample and explicit. Based on the requests the fakes objects are added to data list. Fake objects are objects generated by the distributor 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, in order to increase the chances of detecting agents that leak data.

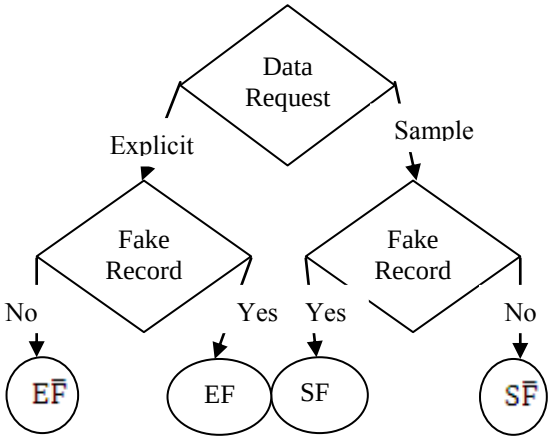

Fig 1: Leakage Problem Instances

The Fig. 1 represents four problem instances with the names *EF*, $E\bar{F}$, *SF* and $S\bar{F}$, where *E* stands for explicit requests, *S* for sample requests, *F* for the use of fake objects, and $\bar{F}$ for the case where fake objects are not allowed.

The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents. Since, fake objects may impact the correctness of what agents do, so they may not always be allowable. Use of fake objects is inspired by the use of "trace" records in mailing lists. The distributor creates and adds fake objects to the data that he distributes to agents. In many cases, the distributor may be limited in how many fake objects he can create

In *EF* problems, objective values are initialized by agents' data requests. Say, for example, that $T = \{t_1, t_2\}$ and there are two agents with explicit data requests such that $R_1 = \{t_1, t_2\}$ and $R_2 = \{t_1\}$. The distributor cannot remove or alter the R_1 or R_2 data to decrease the overlap $R_1 \setminus R_2$. However, say the distributor can create one fake object ($B = 1$) and both agents can receive one

fake object (b_1 = b_2 = 1). If the distributor is able to create more fake objects, he could further improve the objective.

C. *Optimization Problem*

The distributor's data allocation to agents has one constraint and one objective. The distributor's constraint is to satisfy agents' requests, by providing them with the number of objects they request or with all available objects that satisfy their conditions. His objective is to be able to detect an agent who leaks any portion of his data

We consider the constraint as strict. The distributor may not deny serving an agent request and may not provide agents with different perturbed versions of the same objects. The fake object distribution as the only possible constraint relaxation

The objective is to maximize the chances of detecting a guilty agent that leaks all his data objects.

The Pr $\{G_j|S = R_i\}$ or simply Pr $\{G_j \mid R_i\}$ is the probability that agent U_j is guilty if the distributor discovers a leaked table *S* that contains all $R_i$objects. The difference functions
Δ (*i, j*) is defined as:

$$\Delta(i,j) = \Pr\{G_i|R_i\} - \Pr\{G_j|R_i\} \quad \ldots\ldots. 1.5$$

1) *Problem definition:* Let the distributor have data requests from *n* agents. The distributor wants to give tables
R_1, ... R_n to agents $U_1, \ldots, U_n$ respectively, so that

- Distribution satisfies agents' requests; and
- Maximizes the guilt probability differences Δ (*i, j*) for all *i, j* = 1. . . *n* and *i* = *j*.

Assuming that the R_i sets satisfy the agents' requests, we can express the problem as a multi-criterion
2) *Optimization problem:*

Maximize $(\ldots, \Delta(i,j), \ldots)$ *i* != *j* …… 1.6
(*over* $R_1, \ldots, R_n$)

The approximation [3] of objective of the above equation does not depend on agent's probabilities and therefore minimize the relative overlap among the agents as

Minimize $(\ldots, \frac{|R_i \cap R_j|}{R_i}, \ldots)$ *i* != *j* ……..1.7
(*over* $R_1, \ldots, R_n$)

This approximation is valid if minimizing the relative overlap $\frac{|R_i \cap R_j|}{R_i}$ maximizes Δ (*i, j*).

D. *Objective Approximation*

In case of sample request, all requests are of fixed size. Therefore, maximizing the chance of detecting a guilty agent that leaks all his data by minimizing $\frac{|R_i \cap R_j|}{R_i}$ is equivalent to minimizing $(|R_i \cap R_j|)$. The minimum value of $(|R_i \cap R_j|)$ maximizes $\prod(|R_i \cap R_j|)$ and Δ(*i, j*), since $\prod(|R_i|)$ is fixed.

If agents have explicit data requests, then overlaps $|R_i \cap R_j|$ are defined by their own requests and $|R_i \cap R_j|$ are fixed. Therefore, minimizing $|R_i|$ j is equivalent to maximizing $|R_i|$ (with the addition of fake objects). The maximum value of $|R_i|$ mimimizes Π(R_i) and maximizes Δ(*i, j*), since $\Pi(R_i \cap R_j)$ is fixed.

III. Allocation Strategies

In this section the allocation strategies [1] solve exactly or approximately the scalar versions of Equation 1.7 for the different instances presented in Fig. 1. In Section A deals with problems with explicit data requests and in Section B with problems with sample data requests.

A. *Explicit Data Request*

In case of explicit data request with fake not allowed, the distributor is not allowed to add fake objects to the distributed data. So Data allocation is fully defined by the agent's data request.

In case of explicit data request with fake allowed, the distributor cannot remove or alter the requests R from the agent. However distributor can add the fake object. In algorithm for data allocation for explicit request, the input to this is a set of request R_1, R_2,……,R_n from n agents and different conditions for requests. The e-optimal algorithm finds the agents that are eligible to receiving fake objects. Then create one fake object in iteration and allocate it to the agent selected. The e-optimal algorithm minimizes every term of the objective summation by adding maximum number b_i of fake objects to every set R_i yielding optimal solution.

Step 1: Calculate total fake records as sum of fake records allowed.

Step 2: While total fake objects > 0

Step 3: Select agent that will yield the greatest improvement in the sum objective

i.e. i = $\text{argmax}\left(\frac{1}{|R_i|} - \frac{1}{|R_i|+1}\right) \sum_j R_i \cap R_j$

Step 4: Create fake record

Step 5: Add this fake record to the agent and also to fake record set.

Step 6: Decrement fake record from total fake record set.

Algorithm makes a greedy choice by selecting the agent that will yield the greatest improvement in the sum-objective.

B. *Sample Data Request:*

With sample data requests, each agent U_i may

receive any T from a subset out of $\binom{|T|}{m}$ different ones. Hence, there are $\prod_{i=1}^{n}\binom{|T|}{m}$ different allocations. In every allocation, the distributor can permute T objects and keep the same chances of guilty agent detection. The reason is that the guilt probability depends only on which agents have received the leaked objects and not on the identity of the leaked objects. Therefore, from the distributor's perspective there are $\prod_{i=1}^{n}\binom{|T|}{m}/|T|$ different allocations. An object allocation that satisfies requests and ignores the distributor's objective is to give each agent a unique subset of T of size m. The s-max algorithm allocates to an agent the data record that yields the minimum increase of the maximum relative overlap among any pair of agents. The s-max algorithm is as follows.

Step 1: Initialize Min_overlap ← 1, the minimum out of the maximum relative overlaps that the allocations of different objects to U_i

Step 2: for k ∈ {k | $t_k \in R_i$} do

Initialize max_rel_ov ← 0, the maximum relative overlap between $R_i$and any set $R_j$that the allocation of t_k to U_i

Step 3: for all j = 1,..., n : j = i and $t_k \in R_j$do

Calculate absolute overlap as

abs_ov ← $|R_i \cap R_j| + 1$

Calculate relative overlap as

rel_ov ← abs_ov / min (m_i , m_j)

Step 4: Find maximum relative as

max_rel_ov ← MAX (max_rel_ov, rel_ov)

If max_rel_ov ≤ min_overlap then

min_overlap ← max_rel_ov

ret_k ← k

Return ret_k

It can be shown that algorithm s-max is optimal for the sum-objective and the max-objective in problems where M ≤ |T| and n < |T|. It is also optimal for the max-objective if |T| ≤ M ≤ 2 |T| or all agents request data of the same size.

It is observed that the relative performance of algorithm and main conclusion do not change. If p approaches to 0, it becomes easier to find guilty agents and algorithm performance converges. On the other hand, if p approaches 1, the relative differences among algorithms grow since more evidence is need to find an agent guilty.

The algorithm presented implements a variety of data distribution strategies that can improve the distributor's chances of identifying a leaker. It is shown that distributing objects judiciously can make a significant difference in identifying guilty agents, especially in cases where there is large overlap in the data that agents must receive.

IV. Conclusion

In doing a business there would be no need to hand over sensitive data to agents that may unknowingly or maliciously leak it. And even if we had to hand over sensitive data, in a perfect world we could watermark each object so that we could trace its origins with absolute certainty. However, in many cases we must indeed work with agents that may not be 100% trusted, and we may not be certain if a leaked object came from an agent or from some other source. In spite of these difficulties, we have shown it is possible to assess the likelihood that an agent is responsible for a leak, based on the overlap of his data with the leaked data and the data of other agents, and based on the probability that objects can be "guessed" by other means.

Acknowledgment

Students work is incomplete until they thank the almighty & his teachers. I sincerely believe in this and would like to thank Dr. R.J.Anandhi, Head of the Department, Computer Science & Engineering, TOCE, Bangalore for her encouragement and motivation to write this paper. Also I am grateful to Dr. Ravindranath Chowdary, Prof., (CSE), TOCE, Bangalore for guiding me in writing this paper.

References

[1] P. Papadimitriou and H. Garcia-Molina, "Data leakage detection," IEEE Transactions on Knowledge and Data Engineering, pages 51-63, volume 23, 2011.

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

[3] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression, 2002.

[4] S. U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani. Towards robustness in query auditing. In VLDB '06.