IJCSN International Journal of Computer Science and Network, Volume 2, Issue 3, June 2013
ISSN (Online) : 2277-5420 www.ijcsn.org


# Using Unobtrusive Techniques for Detecting Leakage of Data

[1] Shailaja B. Jadhav, [2] Prachi Joshi

[1] Asst. Prof. MMCOE, Pune

[2] Student , MMCOE, Pune

**Abstract**

While doing business, there are situations when some important information or data should be handed over to some third parties. The data important to organization must be secured but the third parties are not that careful about the data. So there is possibility that the data gets leaked. Here in this paper we are considering the chances that the third party is responsible for the leakage. We propose data allocation strategies (across the agents) that improve the probability of identifying leakages. There are some available techniques like watermarking, perturbation used to identifying data leakage, but those are having few drawbacks. So we are trying to detect the leakage by using a technique called unobtrusive. This technique will give us results in many prospective to find out leakage of the important data.

***Keywords:*** *Guilty agent, Data Allocation Problem, Agent Guilt Model, Fake Object, Optimization.*

## 1. Introduction

When we do business, sometimes sensitive data must be handed over to supposedly trusted third parties. For example, a Cricket club may give player records to researchers who will devise new. Similarly, a company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various other companies.

We call the owner of the data the distributor and the supposedly trusted third parties the agents. Our goal is to detect when the distributor's sensitive data have been leaked by agents, and if possible to identify the agent that leaked the data. We consider applications where the original sensitive data cannot be perturbed. Perturbation is a very useful technique where the data are modified and made "less sensitive" before being handed to agents. For example, one can add random noise to certain attributes, or one can replace exact values by ranges. However, in some cases, it is important not to alter the original distributor's data. For example, if outsourcers are doing our payroll, he must have the exact salary and customer bank account numbers. If medical researchers will be treating patients (as opposed to simply computing statistics), they may need accurate data for the patients. Traditionally, leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the misuse can be identified. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermark can sometimes be destroyed if the data recipient is malicious. In this research paper, 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 website, 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. Using an analogy with cookies stolen from a cookie jar, if we catch Freddie with a single cookie, he can argue that a friend gave him the cookie. But if we catch Freddie with five cookies, it will be much harder for him to argue that his hands were not in the cookie jar. If the distributor sees "enough evidence" that an agent leaked data, he may stop doing business with him, or may initiate legal proceedings.

## 2. Problem Setup and Notation

*A. Entities and Agents*

A distributor owns a set T = ft1; t2; t3; : : : g of valuable data objects. The distributor wants to share some of the objects with a set of agents U1; U2; :::; Un, but does not wish the objects be leaked to other third parties. 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. An agent Ui receives a subset of objects Ri T , determined either by a sample request or an explicit request:

- Sample request Ri = SAMPLE (T; mi): Any subset of mi records from T can be given to Ui.
- Explicit request Ri = EXPLICIT (T; condi): Agent Ui receives all the T objects that satisfy condi.

Example: Say T contains customer records for a given company A. Company A hires a marketing agency U1 to do an on-line survey of customers. Since any customers will do for the survey, U1 requests a sample of 1000 customer records. At the same time, company A subcontracts with agent U2 to handle billing for all California customers. Thus, U2 receives all T records that satisfy the condition "state is California."

*B. Guilty Agents*

Suppose that after giving objects to agents, the distributor discovers that a set S 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 U1; : : : ; Un 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. For example, say one of the objects in S represents a customer X. Perhaps X is also a customer of some other company, and that company provided the data to the target. Or perhaps X can be reconstructed from various publicly available sources on the web.

Main goal is to estimate the likelihood that the leaked data came from the agents as opposed to other sources. Intuitively, the more data in S, the harder it is for the agents to argue they did not leak anything. Similarly, the "rarer" the objects, the harder it is to argue that the target obtained them through other means. Not only do we want to estimate the likelihood the agents leaked data, but we would also like to find out if one of them in particular was more likely to be the leaker. For instance, if one of the S objects was only given to agent U1, while the other objects were given to all agents, we may suspect U1 more. The model we present next captures this intuition. We say an agent Ui is guilty if it contributes one or more objects to the target. We denote the event that agent Ui is guilty for a given leaked set S by Gi|jS. Our next step is to estimate P r{Gi jS}, i.e., the probability that agent Ui is guilty given evidence S.

## 3. Agent Guilt Model

To compute this P r{Gi|S}, we need an estimate for 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. We can conduct an experiment and ask a person with approximately the expertise and resources of the target to find the email of say 100 individuals. If this person cans find say 90 emails, then we can reasonably guess that the probability of finding one email is 0.9. On the other hand, if the objects in question are bank account numbers, the person may only discover say 20, leading to an estimate of 0.2. We call this estimate pt, the probability that object t can be guessed by the target. To simplify the formulas that we present in the rest of the paper, we assume that all T objects have the same pt, which we call p. Our equations can be easily generalized to diverse pt's though they become cumbersome to display. Next, we make two assumptions regarding the relationship among the various leakage events. The first assumption simply states that an agent's decision to leak an object is not related to other objects.

Assumption 1: For all t; t' *E* S such that t != t' the provenance of t is independent of the provenance of t'.

To simplify our formulas, the following assumption states that joint events have a negligible probability. As we argue in the example below, this assumption gives us more conservative estimates for the guilt of agents, which is consistent with our goals.

Assumption 2: An object t *E* S can only be obtained by the target in one of two ways:

- A single agent Ui leaked t from his own Ri set; or
- The target guessed (or obtained through other means) t without the help of any of the n agents.

In other words, for all t 2 S, the event that the target guesses t and the events that agent Ui (i = 1; : : : ; n) leaks object tare disjoint.

Before we present the general formula for computing P r{Gi|S}, we provide a simple example. Assume that sets T , R's and S are as follows:

T = {t1; t2; t3}; R1 = {t1; t2}; R2 = {t1; t3}; S = {t1; t2; t3}:

In this case, all three of the distributor's objects have been leaked and appear in S. Let us first consider how the target may have obtained object t1, which was given to both agents. From Assumption 2, the target either guessed t1 or one of U1 or U2 leaked it. We know that the probability of the former event is p, so assuming that the probability that each of the two agents leaked t1 is the same we have the following cases:

- the leaker guessed t1 with probability p;

- agent U1 leaked t1 to S with probability (1 p)=2
- agent U2 leaked t1 to S with probability (1 p)=2

Similarly, we find that agent U1 leaked t2 to S with probability 1 p since it is the only agent that has this data object. Given these values, the probability that agent U1 is not guilty, namely that U1 did not leak either object is:

P r{G1|S} = (1- (1 - p)/2) * (1 - (1- p)) ……….(1)

Hence, the probability that U1 is guilty is: P r{G1|S} = 1 - P r{G1|S}……….(2)
In the general case (with our assumptions), to find the probability that an agent Ui is guilty given a set S, first we compute the probability that he leaks a single object t to S. To compute this we define the set of agents Vt = {Ui|t *E* Ri} that have t in their data sets. Then using Assumption 2 and known probability p, we have:

P r{some agent leaked t to S} = 1 – p…… (3)

Assuming that all agents that belong to Vt can leak t to S with equal probability and using Assumption 2 we 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.

*A. 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 of his data objects. We consider the constraint as strict. The distributor may not deny serving an agent request as in [3] and may not provide agents with different perturbed versions of the same objects as in [4]. We consider fake object allocation as the only possible constraint relaxation.

Our detection objective is ideal and intractable. Detection would be assured only if the distributor gave no data object to any agent. We use instead the following objective: maximize the chances of detecting a guilty agent that leaks all his objects. We now introduce some notation to state formally the distributor's objective. Recall that P r{Gj |S = Ri}or simply P r{Gj |Ri} is the probability that agent Uj is guilty if the distributor obtain:

P r{Ui leaked t to S} = 1-p/V_j; if Ui *E* Vt

0; otherwise…………………….. (4)

Given that agent Ui is guilty if he leaks at least one value to S, with Assumption 1 and Equation 4 we can compute the probability P r{Gi |S} that agent Ui is guilty:

P r{Gi|S} = 1- II ( 1 –(1- p)/|Vt| ………..(5)

## 4. Data Allocation Problem

The main focus of our work is the data allocation problem: how can the distributor "intelligently" give data to agents 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 (E for Explicit and S for Sample requests) and whether "fake objects" are allowed (F for the use of fake objects, and F for the case where fake objects are not allowed).Fake objects are objects generated by discovers a leaked table S that contains all Ri objects.
We define the difference functions delta(i; j) as:

delta(i; j) = P r{Gi|Ri} - P r{Gj |Ri} i; j = 1; : ; n …….(6)

Difference delta (i; j) is positive for any agent Uj , whose set Rj does not contain all data of S. It is zero, if Ri delta Rj. In this case the distributor will consider both agents Ui and Uj equally guilty since they have both received all the leaked objects. The larger a delta (i; j) value is, the easier it is to identify Ui as the leaking agent. Thus, we want to distribute data so that delta values are large:

Problem Definition; Let the distributor have data requests from n agents. The distributor wants to give tables R1; : : : ; Rn to agents U1; : : : ; Un respectively, so that:

- he satisfies agents' requests; and
- he maximizes the guilt probability differences delta(i; j) for all i; j = 1; : : : ; n and i != j.

Assuming that the Ri sets satisfy the agents' requests, we can express the problem as a multi-criterion optimization problem:
Maximize(over R1; : : : ; Rn)(: : : ; (i; j); : : : ) i != j …………………(7)

If the optimization problem has an optimal solution, that means that there exists an allocation D*= {R1*; :

: : ; Rn} such that any other feasible allocation D = {R1; : : : ; Rn} yields delt*(i; j) >=delta(i; j) for all i; j. If there is no optimal allocation D*, a multi- criterion problem has Pareto optimal allocations. If Dpo = {Rpo 1; : : : ; Rpo n }is a Pareto optimal allocation, that means that there is no other allocation that yields delta (i; j) >=delta po (i; j) for all i; j.

*B. Objective Approximation*

We can approximate the objective of Equation 7 with Equation 8 that does not depend on agents' guilt probabilities and therefore on p.

minimize(over R1; : : : ; Rn)=({Ri\R} /|Ri|);i! = j ……………(8)

This approximation is valid if minimizing the relative overlap (Ri\Rj)/ |Ri| maximizes delta(i; j). The intuitive argument for this approximation is that the fewer leaked objects set Rj contains, the less guilty agent Uj will appear compared to Ui (since S = Ri)

In [2] we prove that problems 7 and 8 are equivalent if each T object is allocated to the same number of agents, regardless of who these agents are.We present two different scalar versions of our problem in Equations 9a and 9b. We will refer to objective 9a as the sum objective and to objective 9b as the max- objective.

minimize(over R1; : : : ; Rn)Xni=11jRijXnj=1j=6
ijRi \ Rj j …………..(9a)

minimize(over R1; : : : ; Rn)maxi;j=1;:::;nj=6 ijRi \ Rj
jjRij…………….(9b)

Both scalar optimization problems yield the optimal solution of the problem of Equation 8, if such solution exists. If there is no global optimal solution, the sum-objective yields the Pareto optimal solution that allows the distributor to detect the guilty agent on average (over all different agents) with higher confidence than any other distribution.

The max- objective yields the solution that guarantees that the distributor will detect the guilty agent with a certain confidence in the worst case. Such a guarantee may adversely impact the average performance of the distribution.

## 5. Conclusion

Algorithm 1: Random Fake Object Allocation

Input: R1, . . . , Rn, cond1, . . . , condn, b1, . . . ,bn, B

Output: R1….. Rn, F1,. . . ,Fn

1: R ← null ; Agents that can receive fake objects

2: for i = 1………… n do

3: if bi > 0 then

4: R → R U {i}

5: Fi ← null ; . Set of fake objects given to agent Ui

6: while B > 0 do

7: i ← SELECTAGENTATRANDOM(R; R1; : : : ; Rn)

8: f ← CREATEFAKEOBJECT(Ri; Fi; condi)

9: Ri ← Ri U {f }

10: Fi ← Fi U {f }

11: bi ← bi – 1

12: if bi = 0 then

13: R ← R\{Ri}

14: B ← B – 1

The likelihood that an agent is responsible for a leak is assessed, 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. The algorithms we have presented implement a variety of data distribution strategies that can improve the distributor's chances of identifying a leaker. We have 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.

## References

[1] L. Sweeney, “Achieving k-anonymity privacy protection using generalization and suppression,” River Edge, NJ, USA, pp. 571–588, 2002.

[2] R. Agrawal and J. Kiernan, “Watermarking relational databases,” in VLDB ’02: Proceedings of the 28th international conference on Very Large Data Bases. VLDB Endowment, 2002, pp. 155–166.

[3] S. U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani, “Towards robustness in query auditing,” in VLDB ’06: Proceedings of the 32nd international conference on Very large data bases. VLDB Endowment, 2006, pp. 151–162.

[4] P. Papadimitriou and H. Garcia-Molina, “Data leakage detection,”Stanford University, Tech. Rep., 2008. [Online]. Available: http://dbpubs.stanford.edu/pub/2008-23

[5] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic databases. In Proceedings of the 28th International Conference on Very Large Databases, Hong Kong, PRC, August 2002.

[6] J. Wylie, M. Bigrigg, J. Strunk, G. Ganger, H. Kiliccote, and P. Khosla. Survivable information storage systems. IEEE Computer,33(8):61–68, Aug. 2000.