Guilt Assessment and Fake Data Allocation Strategies for Traitor Detection

G.N.Sowjanya #1,Mrs. N.Nagasubrahmanyeswari #2

#1Student,Aditya Engineering College,Surampalem,East Godavari(Dt)

#2 Asst Professor, Aditya Engineering College,Surampalem,East Godavari(Dt)

Abstract: Data distribution across trusted agents by a source is complicated as there is always the danger of misappropriation by one or more number of users (agents). An alteration to this sensitive data using Watermarks is of no use. In case of a leak there are two possible situations. First the sensitive data is dormant in some individual's device. Second it is most probably published somewhere on the Web. Nothing can be done about the former scenario. We propose data allocation strategies like injecting "realistic but fake" data records to the original sensitive data at random locations such that these fake tuples can be used to identify the leaker among the agents in a latter scenario. We also present algorithms for distributing objects to agents, in a way that improves our chances of identifying a leaker. In proposed system, Brown Robison algorithms are implemented. Using this model a source can assess the likelihood of a traitor among trusted agents in the aftermath of a leak.

***Keywords*—sensitive data, fake objects, data allocation Strategies, Algorithms.**

I.INTRODUCTION

In today' technically empowered data rich environment, it is a major challenge for data holders to prevent data leakage. Loss of large volumes of protected information has become regular headline event, forcing companies to re-issue cards, notify customers and mitigate loss of goodwill from negative publicity. In business, Sometimes sensitive data is handed to trusted third parties. Here, owner of the data is distributor and third party is agent. Main goal of this paper is to detect when the distributor's sensitive data have been leaked by agents and identifying the traitor that leaked the data.

Traditionally, water marking is used for detecting leakages. In watermarking technique, unique code is embedded within the data. But it is not useful with sensitive information as it changes some of bits in data. Also if the recipient is malicious it, may destroy the watermark. Also access control mechanism can be used, that allow only authorized users to access the sensitive data through an access control policies. But it also put restrictions on users and our aim is to provide service to all customers

Later Perturbation technique is used that modifies data and makes it "less sensitive" before being handed to agents. Combined with Fake data allocations and use of unobtrusive techniques for detecting leakage of a set of objects or records like Allocation for Explicit Data Requests, Random Agent Selection fake object allocation , Greedy Selection of agent for optimization, Data allocation, Object Selection etc has significant performance improvements.

In this paper, we implement a model for assessing the "guilt" of agents. We also present Brown-Robinson Algorithm that fixes a tolerance Є> 0 and initializes a certain number of fake tuples for each agent instead of any random agent. Such tuples do not correspond to real entities but appear realistic to the agents. Data allocation strategies are implemented along with proposed algorithms in order to improve the probability of identifying leakages. This new approach increases the chances of finding a traitor compared to the older one.

II RELATED WORK

The data leakage prevention based on the trustworthiness is used to assess the trustiness of the customer. Maintaining the log of all customer's request is related to the data provenance problem i.e. tracing the lineage of objects. The data allocation strategies are concerned, our work is mostly relevant to watermarking that is used as a means of establishing original ownership of distributed objects. Watermarks were initially used in images, video and audio data whose digital representation includes considerable redundancy. If the object to be watermarked cannot be modified, then a watermark cannot be inserted. In such cases, methods that attach watermarks to the distributed data are not applicable.

Finally, there are also different mechanisms to allow only authorized users to access the sensitive information through access control policies, but these are restrictive and may make it impossible to satisfy agent's requests.

III NOTATION

The distributor owns the sensitive data set T= { t1, t2, ………., tn}. The agent Ai request the data objects from distributor. The objects in S could be of any type and size, e.g. they could be tuples in a relation, or relations in a database. The distributor gives the subset of data to each agent. After giving objects to agents, the distributor discovers that a set S of T has leaked. This means some third party has been caught in possession of S. The agent Ai receives a subset Ri of objects T determined either by implicit request or an explicit request.

- Implicit Request Ri = Implicit(T, mi) : Any subset of mi records from T can be given to agent Ai
- Explicit Request Ri = Explicit(T, Condi) : Agent Ai receives all T objects that satisfy Condi

Pr{Gi|S} is the probability that agent Ai is guilty given evidence S. P_t is the probability that object can be guessed by the target.

IV AGENT GULIT MODEL

Let S denote the leaked data set that may be leaked intentionally or guessed by the target user. Since agent having some of the leaked data of S, may be susceptible for leaking the data. But he may argue that he is innocent and that the S data were obtained by target through some other means.

Our goal is to assess the likelihood that the leaked data came from the agents as opposed to other resources. e.g. if one of the object of S was given to only agent A_1, we may suspect A_1 more. So probability that agent A_1 is guilty for leaking data set S is denoted as Pr {G_i | S}.

To compute Pr{Gi|S}, we need an estimate for the probability that values in S can be "guessed" by the target. Probability P_t is analogous to the probabilities used in designing fault-tolerant systems. Two assumptions are made regarding the relationship among the various leakage events:

- For all t; t' ϵ S such that t ≠ t', the provenance of t is independent of the provenance of t'. The term "provenance" in this assumption statement refers to the source of a value t that appears in the leaked set.
- An object t ϵ S can only be obtained by the target in one of the two ways as follows:

(i) A single agent Ai leaked t from its own R_i set.

(ii) The target guessed (or obtained through other means) t without the help of any of the n agents. In other words, for all t ϵ S, the event that the target guesses t and the events that agent Ai (i = 1; . . . ; n) leaks object t are disjoint.

V DATA ALLOCATION PROBLEM

The main focus of this paper is the data allocation problem: how can the distributor "intelligently" give data to agents in order to improve the chances of detecting a guilty agent. The two types of requests : sample and explicit. 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 T objects, in order to increase the chances of detecting agents that leak data.

A) Fake Objects:

The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents. However, fake objects may impact the correctness of what agents do, so they may not always be allowable. The distributor creates and adds fake objects to the data

that he distributes to agents. We let Fi ⊆Ri be the

subset of fake objects that agent Ai receives. Fake objects must be created carefully so that agents cannot distinguish them from real objects.

In many cases, the distributor may be limited in how many fake objects he can create. For example, objects may contain email addresses, and each fake email address may require the creation of an actual inbox (otherwise the agent may discover the object is fake). The inboxes can actually be monitored by the distributor: if email is received form someone other than the agent who was given the address, it is evidence that the address was leaked. Since creating and monitoring email accounts consumes resources, the distributor may have a limit of fake objects.

Similarly, the distributor may want to limit the number of fake objects received by each agent, so as to not arouse suspicions and to not adversely impact the agents activities. Thus, we say that the distributor can send up to b_i fake objects to agent Ai .

(B) 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 condition and 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. We consider fake object distribution 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 .

VI ALLOCATION STRATEGIES

The distributor gives the data to agents such that he can easily detect the guilty agent in case of leakage of data. To improve the chances of detecting guilty agent, he injects fake objects into the distributed dataset. These fake objects are created in such a manner that, agent cannot distinguish it from original objects. One can maintain the separate dataset of fake objects or can create it on demand. In this paper we have used the dataset of fake tuples. Depending upon the addition of fake tuples into the agent's request, data allocation problem is divided into four cases as:

i. Explicit request with fake tuples

ii. Explicit request without fake tuples

iii. Implicit request with fake tuples

iv. Implicit request without fake tuples.

For example, distributor sends the tuples to agents A1 and A2 as R1= {t1, t2} and R2= { t1}. If the leaked dataset is L={ t1}, then agent A2 appears more guilty than A1. So to minimize the overlap, we insert the fake objects in to one of the agent's dataset.

In this paper, we presented the algorithm and the corresponding results for the explicit data allocation with the addition of fake tuples. Whenever any user request for the tuple, it follows the following steps:

1. The request is sent by the user to the distributor.

2. The request may be implicit or explicit.

3. If it is implicit a subset of the data is given.

4. If request is explicit, it is checked with the log, if any previous request is same .

5. If request is same then system gives the data objects that are not given to previous agent.

6. The fake objects are added to agent's request set.

7. Leaked data set L, obtained by distributor is given as an input.

8. Calculate the guilt probability Gi of user.

A) Explicit Data Requests

In the case where we get similar guilt probabilities of the agents, we consider the trust value of agent. These trust values are calculated from the historical behavior of agents. The calculation of trust value is not given here, we just assumed it. The agent having low trust value is considered as guilty agent.

The algorithm for allocation of dataset on agent's explicit request is given below.

```
Algorithm : Brown Robinson Random Algorithm
Input:- i. T={t1,t2,t3,…….tn} -Distributor's Dataset
        ii. R- Request of the agent
        iii. Cond- Condition given by the agent
        iv. m= number of tuples given to an agent
        m<n, selected randomly
Output:- D- Data sent to agent
        1. D=Φ, T'=Φ
        2. For i=1 to n do
        3. If(ti.fields==cond) then
        4. T'=T'U{ ti}
        5. For i=0 to i<m do
        6. D=DU{ ti}
        7. T'=T'-{ ti}
        8. If T'=Φ then
        9. Goto step 2
        10. Allocate dataset D to particular agent
        11. Repeat the steps for every agent
```

To improve the chances of finding guilty agent we can also add the fake tuples to their data sets. Here we maintained the table for duplicate tuples and add randomly these tuples to the agent's dataset.

```
Algorithm2: Addition of fake tuples:
Input: i. D- Dataset of agent
       ii. F- Set of fake tuples
       iii. Cond- Condition given by agent
       iv. b- number of fake objects to be sent
Output:- D- Dataset with fake tuples
       1. While b>0 do
       2. f= select Fake Object at random from set F
       3. D= DU {f}
       4. F= F-{f}
       5. b=b-1
       6. if F=Φ then reinitialize the fake data set.
```

Similarly, we can distribute the dataset for implicit request of agent. For implicit request the subset of distributor's dataset is selected randomly. Thus with the implicit data request we get different subsets. Hence there are different data allocations. An object allocation that satisfies requests and ignores the distributor's objective to give each agent unique subset of T of size m.

B) Implicit Data Requests

Similarly, we can distribute the dataset for implicit request of agent. For implicit request the subset of distributor's dataset is selected randomly. Thus with the implicit data request we get different subsets. Hence there are different data allocations. An object allocation that satisfies requests and ignores the distributor's objective to give each agent unique subset of T of size m. The Brown Robinson 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 Brown Robinson Max algorithm is as follows:

```
1. Initialize Min_Overlap, the minimum out of the
   minimum relative overlaps that the allocations of
   different objects to Ai
2. for k do Initialize max_rel_ov←0, the maximum
   relative overlap between Ri the allocation of tk to
   Ai
3. for all j=1,……,n:j=I and tk ЄRj do calculate
   absolute overlap as abs_ov←calculate relative
   overlap as rel_ov←abs_ov/min(mi, mj)
4. Find maximum relative overlap as
   Max_rel_ov←MAX(max_rel_ov, rel_ov)
      If max_rel_ov≤ min_ov then
      Min_ov←max_rel_ov
```

```
ret_k←k
Return ret_k
```

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.

VII CONCLUSION

Data leakage is a silent type of threat. Your employee as an insider can intentionally or accidentally leak sensitive information. This sensitive information can be electronically distributed via e-mail, Web sites, FTP, instant messaging, spreadsheets, databases, and any other electronic means available – all without your knowledge. To assess the risk of distributing data two things are important, where first one is data allocation strategy that helps to distribute the tuples among customers with minimum overlap and second one is calculating guilt probability which is based on overlapping of his data set with the leaked data set.

VIII REFERENCES

[1]R. Agrawal and J. Kiernan, "Watermarking Relational Databases," Proc. 28th Int'l Conf. Very Large Data Bases (VLDB '02), VLDB Endowment, pp. 155-166, 2002.
[2] P. Buneman and W.-C. Tan, "Provenance in Databases," Proc. ACM SIGMOD, pp. 1171-1173, 2007.
[3] P. Papadimitriou and H. Garcia-Molina, "Data Leakage Detection," technical report, Stanford Univ., 2008.
[4] R. Sion, M. Atallah, and S. Prabhakar, "Rights Protection for Relational Data," Proc. ACM SIGMOD, pp. 98-109, 2003.
Jajodia, P. Samarati, M. L. Sapino, and V. S.
[5] Subrahmanian. Flexible support for multiple access control policies. ACM Trans. Dataset Systems, 26(2):214-260,2001.
[6] L. Sweeney, "Achieving K-Anonymity Privacy Protection Using Generalization and Suppression," http://en.scientificcommons.org/43196131, 2002.
[7] Y. Cui and J. Widom, "Lineage Tracing for General Data Warehouse Transformations," The VLDB J., vol. 12, pp. 41-58, 2003.
[8] L. Sweeney, "Achieving K-Anonymity Privacy Protection Using Generalization and Suppression," http://en.scientificcommons.org/43196131,2002.