ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

Prevention of Information Leakage

E.Suganthi[1], M.Pushpalatha[2], N.Hemalatha[3]

U.G. Student, Department of Information Technology, Paavai Engineering College, Pachal Nagar, Tamil Nadu, India[1].

Associate Professor, Department of Information Technology, Paavai Engineering College, Pachal Nagar, India[2].

Associate Professor, Department of Information Technology, Paavai Engineering College, Pachal Nagar, India[3].

ABSTRACT: A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data are leaked and found in an unauthorized place (e.g., on the web or somebody's laptop). The distributor must assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means. We propose data allocation strategies (across the agents) that improve the probability of identifying leakages. These methods do not rely on alterations of the released data (e.g., watermarks). 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.

KEYWORDS: Allocation strategies, data leakage, data privacy, fake records, leakage model.

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 record store searchers who will devise new treatments. 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 [18]. However, in some cases, it is important not to alter the original distributor's data. For example, if an outsourcer is 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 leaker can be identified. Watermark scan be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermark scan sometimes be destroyed if the data recipient is malicious. In this 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. In this paper, 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 leaker. Finally, we also consider the option of adding "fake" objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects act as a type of watermark for the entire set, without modifying any individual members. If it turns out that an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

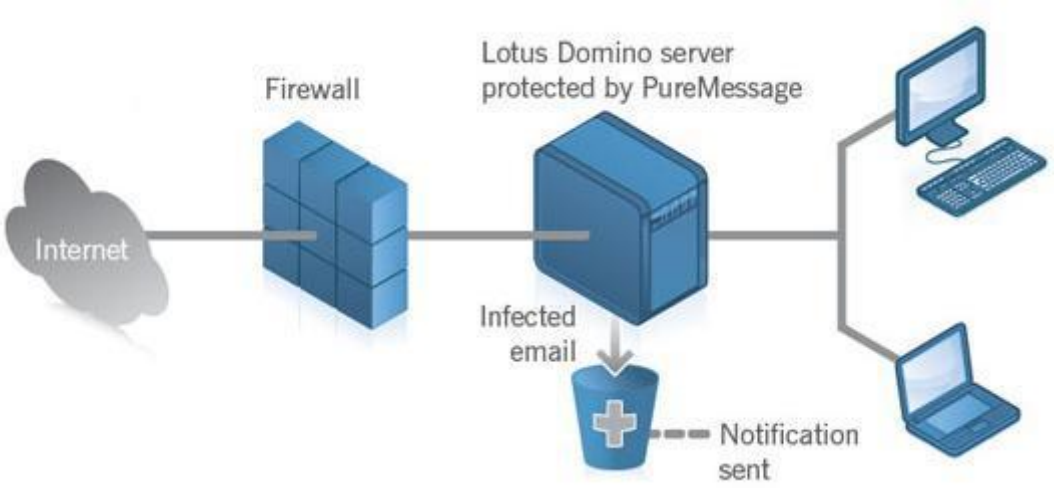
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

II. DATA LEAKAGE DETECTION

Our goal is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data. Perturbation is a very useful technique where the data is modified and made "less sensitive" before being handed to agents. We develop *unobtrusive* techniques for detecting leakage of a set of objects or records.In this section 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 leaker. Finally, we also consider the option of adding "fake" objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects acts as a type of watermark for the entire set, without modifying any individual members. If it turns out an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.



Problem Setup and Notation

A distributor owns a set $T=\{t_{1,\ldots,}t_m\}$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 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, determined either by a sample request or an explicit request:

1. *Sample request*
2. *Explicit request*

Guilt Model Analysis

our model parameters interact and to check if the interactions match our intuition, in this section we study two simple scenarios as **Impact of Probability** p and **Impact of Overlap between** Ri **and** S. In each scenario we have a target that has obtained all the distributor's objects, i.e., T = S.

Algorithms

1.Evaluation of Explicit Data Request Algorithms

In the first place, the goal of these experiments was to see whether fake objects in the distributed data sets yield significant improvement in our chances of detecting a guilty agent. In the second place, we wanted to evaluate our e-optimal algorithm relative to a random allocation.

2. Evaluation of Sample Data Request Algorithms

With sample data requests agents are not interested in particular objects. Hence, object sharing is not explicitly defined by their requests. The distributor is "forced" to allocate certain objects to multiple agents only if the number of

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

requested objects exceeds the number of objects in set T. The more data objects the agents request in total, the more recipients on average an object has; and the more objects are shared among different agents, the more difficult it is to detect a guilty agent.

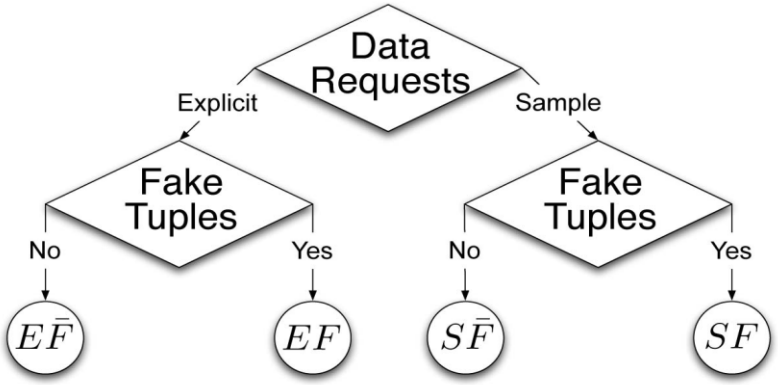
PREVENTION OF INFORMATION LEAKAGE METHODS

DATA ALLOCATION PROBLEM

How can the distributor "intelligently" give data to agents in order to improve the chances of detecting a guilty agent? As illustrated in 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 in 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. That simplicity, we are assuming that in the E problem instances, all agents make explicit requests, while in the S instances, all agents make sample requests. Our results can be extended to handle mixed cases, with someexplicit and some sample requests. We provide here a small example to illustrate how mixed requests can be handled, but then do not elaborate further.

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 alwaysbe allowable. The idea of perturbing data to detect leakage is not new, e.g., [1]. However, in most cases, individual objects are perturbed, e.g., by adding random noise to sensitive salaries, or adding a watermark to an image. In our case,fake elements. In some applications, fake objects may cause fewer problems that perturbing real objects. For example, say that the distributed data objects are medical records and the agents are hospitals. In this case, even small modifications to the records of actual patients may be undesirable. However, the addition of some fake medical records may be acceptable, since no patient matches these records, and hence, no one will ever be treated based on fake records. Our use of fake objects is inspired by the use of "trace" records in mailing lists. In this case, company A sells to company B a mailing list to be used once (e.g., to send advertisements). Company A adds trace records that contain addresses owned by company A. Thus, each time company B uses the purchased mailing list, A receives copies of the mailing. These records are a type of fake objects that help identify improper use of data.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 Ui receives. As discussed below, 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 e-mail addresses, and each fake e-mail address may require the creation of an actual inbox (otherwise, the agent may discover that the object is fake). The inboxes can actually be monitored by the distributor: if e-mail is received from someone other than the agent who was given the address, it is evident that the address was leaked. Since creating and monitoring e-mail accounts consumes resources, the distributor may have a limit of fake objects. If there is a limit, we denote it by B 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 bi fake objects to agent Ui.



ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015

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 satisfytheir 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 as in and may not provide agents with different perturbed versions of the same objects as in . 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 (Mungamuru and Garcia-Molina discuss that to attain "perfect" privacy and security, we have to sacrifice utility). We use instead the following objective.

DATA DISTRIBUTOR

A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data is leaked and found in an unauthorized place (e.g., on the web or somebody's laptop). The distributor must assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means.Admin can able to view the which file is leaking and fake user's details also procedures.

III. RESULTS OF INFORMATION LEAKAGE DETECTION

With sample data requests, agents are not interested in particular objects. Hence, object sharing is not explicitlydefined by their requests. The distributor is "forced" to allocate certain objects to multiple agents only if the numberof requested objects Pn i¼1 mi exceeds the number of objects in set T. The more data objects the agents request in total, the more recipients, on average, an object has; and the more objects are shared among different agents, the more difficult it is to detect a guilty agent. Consequently, the parameter that primarily defines the difficulty of a problem with sample data requests is the ratio Pn.We call this ratio the load.

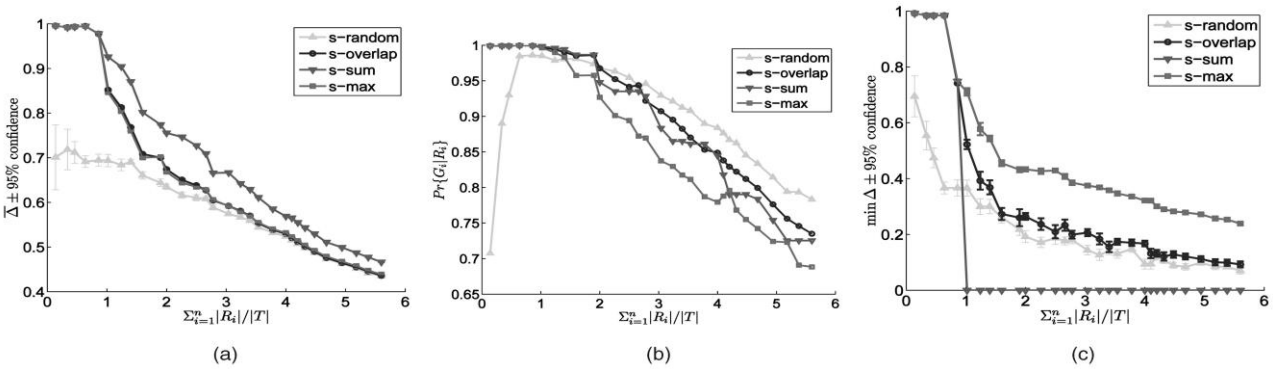
In our experimental scenarios, set T has 50 objects and we vary the load. There are two ways to vary this number: 1) assume that the number of agents is fixed and vary their sample sizes mi, and 2) vary the number of agents whorequest data. The latter choice captures how a real problem may evolve. The distributor may act to attract more or fewer agents for his data, but he does not have control upon agents' requests. Moreover, increasing the number of agents allows us also to increase arbitrarily the value of the load, while varying agents' requests poses an upper bound njTj. Our first scenario includes two agents with requests m1 and m2 that we chose uniformly at random from the interval 6; . . . ; 15. For this scenario, we ran each of the algorithms s-random (baseline), s-overlap, s-sum, and s-max 10 different times, since they all include randomized steps. For each run of every algorithm, we calculated _ andmin_ and the average over the 10 runs. The secondscenario adds agent U3 with m3 _ U½6; 15_ to the two agents of the first scenario. We repeated the 10 runs for each algorithm to allocate objects to three agents of the second scenario and calculated the two metrics values for each run. We continued adding agents and creating new scenarios to reach the number of 30 different scenarios. The last one had 31 agents. Note that we create a new scenario by adding an agent with a random request mi 15 instead of assuming mi ¼ 10 for the new agent. We did that to avoid studying scenarios with equal agent sample request sizes, where certain algorithms have particular properties,
e.g., s-overlap optimizes the sum-objective if requests are all the same size, but this does not hold in the general case.

ISSN(Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Special Issue 6, May 2015



IV. CONCLUSIONS

IN a perfect world, 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 percent trusted, and we may not be certain if a leaked object came from an agent or from some other source, since certain data cannot admit watermarks In spite of these difficulties, we have shown that 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.Our model is relatively simple, but we believe that it captures the essential trade-offs. 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.

Our future work includes the investigation of agent guilt models that capture leakage scenarios that are not studied in this paper. For example, what is the appropriate model for cases where agents can collude and identify fake tuples A preliminary discussion of such a model is available in Another open problem is the extension of our allocation strategies so that they can handle agent requests in an online fashion (the presented strategies assume that there is a fixed set of agents with requests known in advance).

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. Bonatti, S.D.C. di Vimercati, and P. Samarati, "An Algebra forComposing Access Control Policies," ACM Trans. Information and System Security, vol. 5, no. 1, pp. 1-35, 2002.
[3] P. Buneman, S. Khanna, and W.C. Tan, "Why and Where: ACharacterization of Data Provenance," Proc. Eighth Int'l Conf.Database Theory (ICDT '01), J.V. den Bussche and V. Vianu, eds.,pp. 316-330, Jan. 2001.
[4] P. Buneman and W.-C. Tan, "Provenance in Databases," Proc.ACM SIGMOD, pp. 1171-1173, 2007.
[5] Y. Cui and J. Widom, "Lineage Tracing for General DataWarehouse Transformations," The VLDB J., vol. 12, pp. 41-58,2003.
[6] S. Czerwinski, R. Fromm, and T. Hodes, "Digital Music Distributionand Audio Watermarking," http://www.scientificcommons.org/43025658, 2007.
[7] F. Guo, J. Wang, Z. Zhang, X. Ye, and D. Li, "An ImprovedAlgorithm to Watermark Numeric Relational Data," Information Security Applications, pp. 138-149, Springer, 2006.
[8] F. Hartung and B. Girod, "Watermarking of Uncompressed and Compressed Video," Signal Processing, vol. 66, no. 3, pp. 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 Systems, vol. 26, no. 2, pp. 214-260, 2001.