ISSN 2278-8808
An International Peer Reviewed
SCHOLARLY RESEARCH JOURNAL FOR INTERDISCIPLINARY STUDIES

DATA LEAKAGE DETECTION IN CLIENT-SERVER MODEL ON A NETWORK WIRE IN CLIENT-SERVER

Rahul. R. Patil
Assistant Professor Computer science,D. A. B. N. College,

Vilas. S. Patil
Assistant Professor in Physics,Y. C. W. M. Waranagar, Tal-Panhala, Dist- Kolhapur.

Chandrakant B. Mane
Assistant Professor in Chemistry,V. Y. M. Pethvadgaon. Tal-Hatkalagane, Dist-Kolhapur.
Shankar. K. Patil
Assistant Professor in Chemistry,D. A. B. N. College, Chikhli
Manisha V. Patil.
Librarian, Shripatrao Chougule Arts & Commerce Mahavidyalaya, Malwadi-Kotoli. Tal- Panhala.Dist- Kolhapur.

Research Guide :
Manik Kadam
Head of Dept.(Computer Management)Jaywant Institution of Management,,Pune

Abstract

Modern business activities rely on extensive email exchange. Email leakages have become widespread, and the severe damage caused by such leakages constitutes a disturbing problem for organizations. We study the following problem: 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. The distributor must assess the likelihood that the leaked came from one or more agents, as opposed to having been independently gathered by other means. 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. We propose data allocation strategies that improve the probability of identifying leakages. In some cases we can also inject "realistic but fake" data record to further improve our changes of detecting leakage and identifying the guilty party. 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.*

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 part- nerships 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. Per- turbation 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 at- tributes, 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 num- bers. 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 em- bedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, 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.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.

METHODOLOGY:

In this paper, we presented the algorithm and the corresponding results for the explicit data allocation with the addition of fake tuples. We are still working on minimizing the overlap in case of implicit request. 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 using II. 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.

```
a. Algorithm1: Allocation of Data Explicitly: 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(t .fields==cond) then
4. T'=T'U{ t i}
```

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.

b. 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. 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:

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. Fi nd 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

Instrument for Data Collection:

InterviewsIn Quantitative research (survey research), interviews are more structured than in Qualitative research

Questionnaires

Paper-pencil-questionnaires can be sent to a large number of people and saves the researcher time and money. People are more truthful while responding to the questionnaires regarding controversial issues in particular due to the fact that their responses are anonymous. But they also have drawbacks. Majority of the people who receive questionnaires don't return them and those who do might not be representative of the originally selected sample.

Drafting a questionnaire:

The processes of developing questions begin from as there are several critical questions of which evaluation needs to answer. The importance of exact wording in each question is very significant. A great deal of research has studied the effects of question wording and style on responses. While writing good questions may seem to be more of an art than a science, some basic principles for writing questions can serve as a guide for developing a written instrument. Of all the data collection methods questionnaires is a widely used method of collecting information. They can be a cost effective way to reach a large number of people or a geographically diverse group.

Conclusion:

We have shown it is possible to assess the likelihood that an agent is respon- sible 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 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 appro- priate model for cases where agents can collude and identify fake tulles? 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).

BIBLIOGRAPHY

1. R. Agawam and J. Kiernan. Watermarking relational databases. In VLDB ’02: Proceedings of the 28th international conference on Very Large Data Bases, pages 155–166. VLDB Endowment, 2002.
2. P. Bonita, S. D. C. did Vimercati, and P. Samarati. Algebra for composing access control policies. ACM Trans. Inf. Syst. Secur., 5(1):1–35, 2002.
3. 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.
4. Y.Cui and J. Widom. Lineage tracing for general data warehouse transformations. In The VLDB Journal, pages 471–480, 2001.
5. B. Mungamuru and H. Garcia-Molina. Privacy, preservation and performance: The 3 p’s of distributed data management. Technical report, Stanford University, 2008.

Sites Referred:

1. http://www.sourcefordgde.com

2. http://www.networkcomputing.com/

3. http://www.ieee.org