Assess Agent Guilt Model and Handling Data Allocation Strategies for Data Distribution

S. Poovarasan
Student
Department of Computer Science & Engineering
Knowledge Institute of Technology, Salem

C. Sakthivel
Student
Department of Computer Science & Engineering
Knowledge Institute of Technology, Salem

A.Velkuppannasamy
Student
Department of Computer Science & Engineering
Knowledge Institute of Technology, Salem

Mr.T. Karthikeyan
Assistant Professor
Department of Computer Science & Engineering
Knowledge Institute of Technology, Salem

Abstract

A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). 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 have independent that gathered by other means. Data allocation strategies that we proposed (across the agents) improve the probability of identifying leakages. The model doesn't rely on alterations of the released data (e.g., watermarks). In the proposed system, injected "realistic but fake" data records to further improve the chances of detecting leakage and identifying the guilty party.

Keywords: Guilt Model, Watermarking, Data distribution, Perturbation

I. INTRODUCTION

While 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. Another enterprise may outsource its data processing, so data must be given to various other companies. The owner of the data is said to be the distributor and the supposedly trusted third parties the agents. Aim of the work 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.

II. EXISTING APPROACH

A. Perturbation:

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. In some cases, it is important not to alter the original distributor's data. For example, payroll, the exact salary and customer bank account numbers may be modified.

B. Watermarking:

In watermarking, a unique code is embedded in each 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.

III. ADVANCED APPROACH

Unobtrusive techniques for detecting leakage of a set of objects or records have been studied. 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. Using an analogy with cookies stolen from a cookie jar, if Freddie with a single cookie has been cached, he can argue that a friend gave him the cookie. But if Freddie with 5 cookies has been cached, 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.

A model for assessing the "guilt" of agents has been developed. An algorithm for distributing objects to agents, in a way that improves our chances of identifying a leaker has been proposed. The option of adding "fake" objects to the distributed set also been considered. 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

IV. MODULE DESCRIPTION

A. Data Allocation:

This module is mainly designed to transfer data from distributor to agents. The same module can also be used for illegal data transfer from authorized to agents to other agents

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 can be addressed, depending on the type of data requests made by agents and whether "fake objects" are allowed. The two types of requests handled are: sample and explicit. Fake objects are objects generated by the distributor.

The objects are designed to look like real objects, and are distributed to agents together, in order to increase the chances of detecting agents that leak data.

All agents make explicit requests and all agents make sample requests. The results can be extended to handle mixed cases, with some explicit and some sample requests.

B. GUILT Model:

This module is designed using the agent – guilt model. When the distributor sends data to agent, while run time this module allocates unique fake object in each and every tuple provided to agents. A copy of data, which is transferred to agents are stored in the distributor's database. Distributor adds fake objects to the distributed data in order to improve the effectiveness in detecting guilty agents. However, fake objects may impact the correctness of what agents do, so they may not always be allowable.

In this module, perturbing the set of distributor objects by adding fake elements has been done. In some applications, fake objects may cause fewer problems that perturbing real objects. For example, say the distributed data objects are patient records In this case; even small modifications to the records of actual patient distribution records may be undesirable. However, the addition of some fake object may be acceptable.

C. AGENT-GUILT Model:

This module is mainly designed for determining fake agents. This module uses fake objects (which is stored in database from guilt model module) and determines the guilt agent along with the probability. Once the distributor finds his data in unauthorized places, he can able to compare the release data with his copy of data, which is distributed to agents, then he can able to find out the guilt agent. To compute this probability, we need an estimate for the probability that values can be "guessed" by the target. We use the probability of guessing to identify agents that have leaked information. The probabilities are estimated based on experiments.

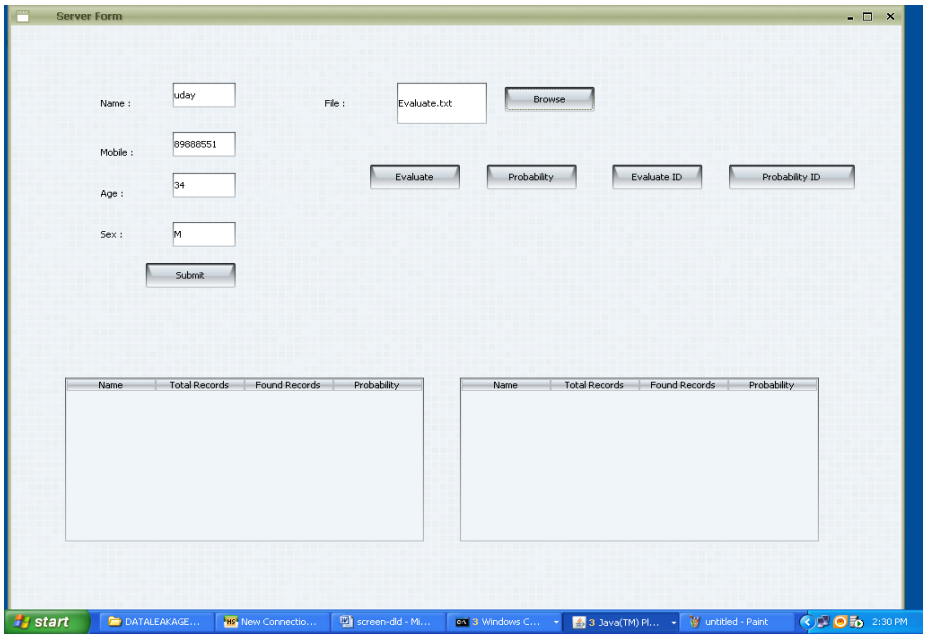

Fig. 1: AGENT-GUILT Model

V. OVERLAP MINIMIZATION

When the distributor allocates data upon request from agents, there may be chance of asking same tuples by more that one agent. Here come overlap between more that one agent, while giving same tuple to more than one agent. When the data is released, distributor must able to assess who is guilt agent.

Thus we are arrive solution for this overlap minimization. When the distributor allocates data along with fake object, the fake objects are should be in the manner that for each and every agent and for each and every tuple the fake object is unique.

VI. FINDING PROBABILITY

When the distributor finds the leaked data in unauthorized places like websites or laptop, then the distributor must be able to find out who is the guilt agent. For this, he can able to find the probability of the number of allocated data for a particular agent by number of found records in unauthorized place. From this it is clearly concluded that the agent who is having more probability must be the guilt agent. Fake objects are used here to confirm the guilt agent more clearly.

VII. 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 idea of perturbing data to detect leakage is not new. 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, we are perturbing the set of distributor objects by adding 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. The 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. 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. 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. The creation of fake but real-looking objects is a nontrivial problem whose thorough investigation is beyond the scope of this paper.

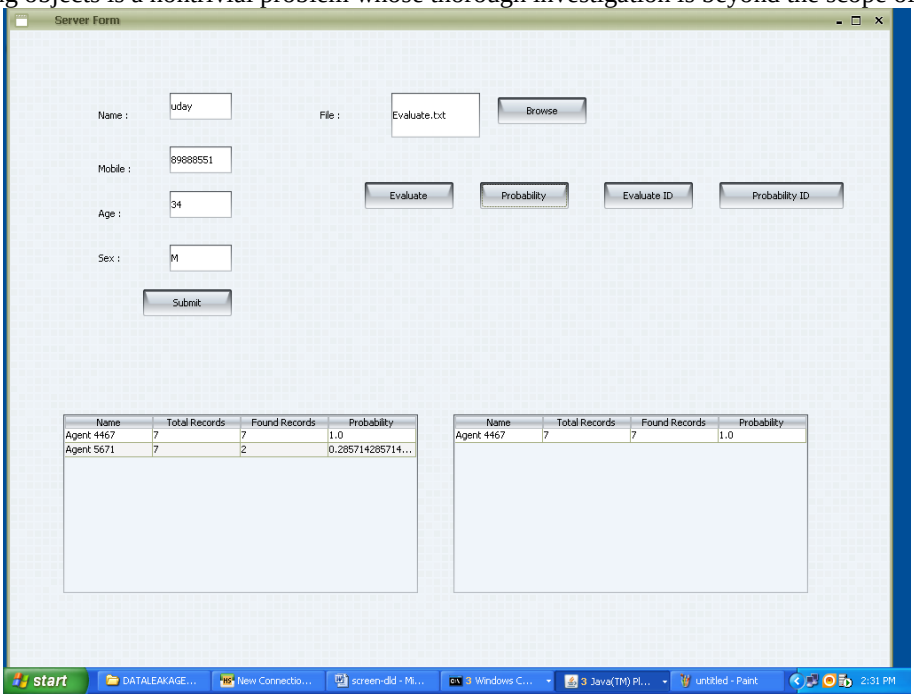

Fig. 2: Fake Objects

VIII. FUTURE ENHANCEMENT

Future work may include the appropriate model for cases where agents can collude and identify fake tuples. 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).

IX. CONCLUSION

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] R. Agrawal 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. Bonatti, S. D. C. di Vimercati, and P. Samarati. An algebra for composing access control policies. ACM Trans. Inf. Syst. Secur., 5(1):1–35, 2002.
[3] P. Buneman, S. Khanna, and W. C. Tan. Why and where: A characterization of data provenance. In J. V. den Bussche and V. Vianu, editors, Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings, volume 1973 of Lecture Notes in Computer Science, pages 316–330. Springer, 2001.
[4] 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.
[5] Y. Cui and J. Widom. Lineage tracing for general data warehouse transformations. In The VLDB Journal, pages 471–480, 2001.
[6] S. Czerwinski, R. Fromm, and T. Hodes, "Digital Music Distribution and Audio Watermarking," http://www.scientificcommons. org/43025658, 2007.
[7] F. Guo, J. Wang, Z. Zhang, X. Ye, and D. Li, "An Improved Algorithm 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.
[10] Y. Li, V. Swarup, and S. Jajodia, "Fingerprinting Relational Databases: Schemes and Specialties," IEEE Trans. Dependa