ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, June 2012

A Model for identifying Guilty Agents in Data Transmission

Shreyta Raj , Dr. Ravinder Purwar , Ashutosh Dangwal

Abstract-- **This research paper presents a formal method for representing and detecting inconsistencies of combined secrecy models is to detect when the PC distributor's sensitive data has been leaked by their agents, and if possible to identify the agent that leaked the data. Data leakage is a silent type of threat. 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. Data allocation strategies (across the agents) are proposed that improve the probability of identifying leakages. These methods do not rely on alterations of the released data (e.g., watermarks). In some cases the distributor can also inject "realistic but fake" data records to further improve our chances of detecting leakage and identifying the guilty party. A model for assessing the "guilt" of agents using C# dot net technologies with MS sql server as backend is proposed to develop. Algorithms for distributing objects to agents, in a way that improves our chances of identifying a leaker is aloes presented. Finally, the option of adding "fake" objects to the distributed set is also considered. Such objects do not correspond to real entities but appear.**

Index Terms - **Data Leakage, Data Privacy, Fake Record,**.

I. INTRODUCTION

While doing business, practical necessities may motivate the use of secrecy models in combination and in addition other business policies may be necessary 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. 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 has 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 is 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 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. 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 work, unobtrusive techniques for detecting leakage of a set of objects or records have been studied. For example, after giving a set of objects to agents, the distributor discovers some of those same objects in an unauthorized place. 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 5 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 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.

II. EXISTING SYSTEM

A. Perturbation

Data perturbation refers to a data transformation process typically performed by the data owners before publishing their data. The goal of performing such data transformation is two-fold. On one hand, the data owners want to change the data in a certain way in order to disguise the sensitive information contained in the published datasets, and on the other hand, the data owners want the transformation to best preserve. 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 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.

B. Watermarking

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. 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.

III. PROPOSED SYSTEM

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.

Advantages-

- After giving a set of objects to agents, the distributor discovers some of those same objects in an unauthorized place.

- 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.

- If the distributor sees “enough evidence” that an agent leaked data, he may stop doing business with him, or may initiate legal proceedings.

- To 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.

- Consider the option of adding “fake” objects to the distributed set. Such objects do not correspond to real entities but appear.

- 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. IMPLEMENTATION PLAN

In this application, we try to implement a model application to detect the data leakages between distributor and agents. The system is developed in C# dot net. We use Microsoft SQL Server 2000 as database for this application. When the distributor sends a file to agent, it is considered as fake object for that particular agent, in this sequence file name and file path is stored in the database for future reference.

Similarly when the agent sends a file to the unauthorized agent the sequence is store in the database. Thus we can find the guilt agent. The probability function is calculated based on the number of guilt agents by the number of file transfers between the agent and unauthorized person.

A. MODULE DESCRIPTION

1- Login / Registrations - This is a module mainly designed to provide the authority to a user in order to access the other modules of the project.
2- Data Transfer- 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
3- Guilt Model Analysis- This module is designed using the agent – guilt model. Here a count value(also called as fake objects) are incremented for any transfer of data occurrence when agent transfers data. Fake objects are stored in database.
4- 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. A graph is used to plot the probability distribution of data which is leaked by fake agents.

To compute this probability, we need an estimate for the probability that values can be “guessed” by the target.

B. Algorithm Steps

Step: 1 Distributor select agent to send data

The distributor selects two agents and gives requested data R1, R2 to both agents.

Step: 2 Distributor creates fake object and allocates it to the agent

The distributor can create one fake object (B = 1) and both agents can receive one fake object (b1 = b2 = 1). If the distributor is able to create more fake objects, he could further improve the objective.

Step: 3 check number of agents, who have already received data

Distributor checks the number of agents, who have already received data.

Step: 4 Check for remaining agents

Distributor chooses the remaining agents to send the data. Distributor can increase the number of possible allocations by adding fake object.

Step: 5 Select fake object again to allocate for remaining agents

Distributor chooses the random fake object to allocate for the remaining agents.

Step: 6 Estimate the probability value for guilt agent To compute this probability, we need an estimate for the probability that values can be "guessed" by the target.

V. WHY USE DATAMINING

Data mining is the process of extracting patterns from data. Data mining is becoming an increasingly important tool to transform the data into information. It is commonly used in a wide range of profiling practices such as marketing, surveillance fraud detection and scientific discovery. Data mining can be used to uncover patterns in data but is often carried out only on samples of data. The mining process will be ineffective if the samples are not a good representation of the larger body of data. Data mining cannot discover patterns that may be present in the larger body of data if those patterns are not present in the sample being "mined". Inability to find patterns may become a cause for some disputes between customers and service providers. Therefore data mining is not foolproof but may be useful if sufficiently representative data samples are collected. The discovery of a particular pattern in a particular set of data does not necessarily mean that a pattern is found elsewhere in the larger data from which that sample was drawn. An important part of the process is the verification and validation of patterns on other samples of data

VI. LITERATURE REVIEW

The guilt detection approach we present is related to the data provenance problem, tracing the lineage of S objects implies essentially the detection of the guilty agents.

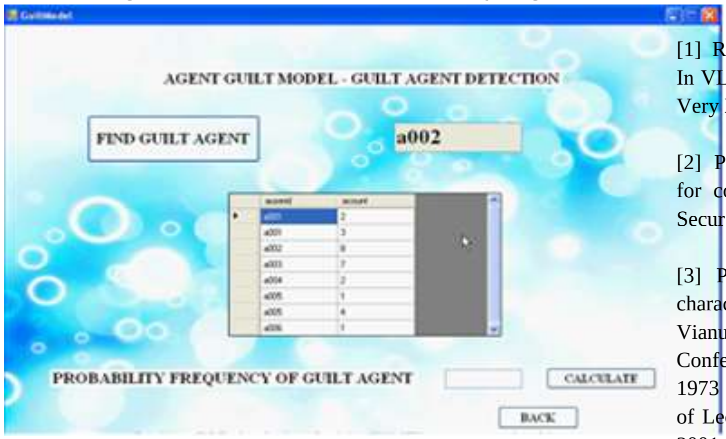
- Suggested solutions are domain specific, such as lineage tracing for data warehouses and assume some prior knowledge on the way a data view is created out of data sources.

- Watermarks were initially used in images, video and audio data whose digital representation includes considerable redundancy. Watermarking is similar in the sense of providing agents with some kind of receiver-identifying information. However, by its very nature, a watermark modifies the item being watermarked. 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.

- Recently, works have also studied marks insertion to relational data.

- There are also lots of other works on mechanisms that allow only authorized users to access sensitive data through access control policies. Such approaches prevent in some sense data leakage by sharing information only with trusted parties. However, these policies are restrictive and may make it impossible to satisfy agents' requests.

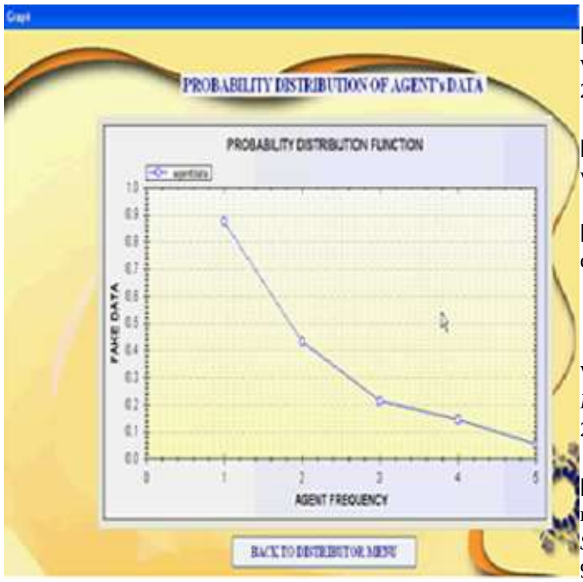
Algorithm Results to find out Guilty Agents


GuiltModel
AGENT GUILT MODEL - GUILT AGENT DETECTION
FIND GUILT AGENT
a002
PROBABILITY FREQUENCY OF GUILT AGENT
CALCULATE
BACK

[illegible]	[illegible]
[illegible]	2
a001	3
a002	8
a003	7
a004	2
a005	1
a005	4
a006	1

Note: a002 is the guilty agent.

Probability distribution of Agent Data



7. 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] V. N. Murty. Counting the integer solutions of a linear equation with unit coefficients. *Mathematics Magazine*, 54(2):79–81, 1981.

[7] P. Papadimitriou and H. Garcia-Molina. Data leakage detection.Technical report, Stanford University, 2008.

[8] J. J. K. O. Ruanaidh, W. J. Dowling, and F. M. Boland. Watermarking digital images for copyright protection. *I.E.E. Proceedings on Vision, Signal and Image Processing*, 143(4):250–256, 1996.

[9] R. Sion, M. Atallah, and S. Prabhakar. Rights protection for relational data. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 98–109, New York, NY, USA, 2003. ACM.

[10] L. Sweeney. Achieving k-anonymity privacy protection usinggeneralization and suppression, 2002.

[11] 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*, pages 151–162. VLDB Endowment, 2006.

[12] P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is np-hard. *Journal of Global Optimization*, 1(1):15–22, 1991.