Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 13 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Instructive of Ooze Information

By K. Sudheer Kumar & Ch.S.V.V.S.N.Murthy
Sri Sai Aditya Institute of Science And Technology, Surampalem, Kakinada, Andhra Pradesh, India

Abstract - We study the following problem: A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data are leaked and bring into being in an unconstitutional place (e.g., on the web or somebody's laptop). The distributor must evaluate 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 distribution strategies (across the agents) that improve the likelihood 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 replica" data records to further improve our chances of detecting leakage and identifying the guilty party.

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 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. There always remains a risk of data getting leaked from the agent.

Perturbation is a very valuable 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. But this technique requires modification of data. Leakage detection is handled by watermarking, e.g., a unique code is implanted in each distributed copy. If that copy is later discovered in the hands of an unconstitutional party, the leaker can be identified. But again it requires code modification. Watermarks can sometimes be destroyed if the data recipient is malicious.

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

GJCST-C Classification: E.0

INSTRUCTIVE OF OOZE INFORMATION

Strictly as per the compliance and regulations of:

Instructive of Ooze Information

K. Sudheer Kumar [α] & Ch. S.V.V.S.N.Murthy [σ]

Abstract - We study the following problem: A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data are leaked and bring into being in an unconstitutional place (e.g., on the web or somebody's laptop). The distributor must evaluate 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 distribution strategies (across the agents) that improve the likelihood 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 replica" data records to further improve our chances of detecting leakage and identifying the guilty party.

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 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. There always remains a risk of data getting leaked from the agent.

Perturbation is a very valuable 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. But this technique requires modification of data. Leakage detection is handled by watermarking, e.g., a unique code is implanted in each distributed copy. If that copy is later discovered in the hands of an unconstitutional party, the leaker can be identified. But again it requires code modification. Watermarks can sometimes be destroyed if the data recipient is malicious.

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 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 [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. 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 (Not attracting unnecessary attention) 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 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. 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.

Author α : Department of Computer Science & Engineering, Sri Sai Aditya Institute of Science And Technology, Surampalem, Kakinada, Andhra Pradesh, India. E-mail : sudheerkumarkotha@gmail.com
Author σ : Department of Information Technology, Sri Sai Aditya Institute of Science And Technology, Surampalem, Kakinada, Andhra Pradesh, India. E-mail : chsatyamurty@gmail.com

II. Problem Setup and Notation

Entities and Agents: A distributor owns a set T={t1, tm} of valuable data objects. The distributor wants to share some of the objects with a set of agents U1, U2, Un, 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 Ri ⊆ T, determined either by a sample request or an explicit request:

Sample request Ri = SAMPLE (T, mi): Any subset of Mi records from T can be given to Ui.

Explicit request Ri = EXPLICIT (T, condi): Agent Ui receives all the T objects that satisfy condi.

Type of data leakage: In order to implement the appropriate protective measures, we must first understand what we are protecting. Based on publicly disclosed Data Leakage breaches, the type of data leaked is broken down as follows:

Type of information leaked Percentage
Confidential information - 15%
Intellectual property - 4%
Customer data - 73%
Health records - 8%

Guilty Agents: Suppose that after giving objects to agents, the distributor discovers that a set S ⊆ T has leaked. This means that some third party called the target has been caught in possession of S. For example, this target may be displaying S on its web site, or perhaps as part of a legal discovery process, the target turned over S to the distributor.

Since the agents U1, Un has some of the data, it is reasonable to suspect them leaking the data. However, the agents can argue that they are innocent, and that the S data was obtained by the target through other means.

For example, say one of the objects in S represents a customer X. Perhaps X is also a customer of some other company, and that company provided the data to the target. Or perhaps X can be reconstructed from various publicly Available sources on the web.

Our goal is to estimate the likelihood that the leaked data came from the agents as opposed to other sources. Intuitively, the more data in S, the harder it is for the agents to argue they did not leak anything. Similarly, the "rarer" the objects, the harder it is to argue that the target obtained them through other means. For instance, if one of the S objects was only given to agent U1, while the other objects were given to all agents, we may suspect U1 more. The model we present next captures this intuition.

We say an agent Ui is guilty and if it contributes one or more objects to the target. We denote the event that agent Ui is guilty as Gi and the event that agent Ui is guilty for a given leaked set S as Gi|S. Our next step is to estimate Pr {Gi|S}, i.e., the probability that agent Ui is guilty given evidence S.

III. Related Works

The guilt detection approach we present is related to the data provenance problem [3]: (whether it is genuine or not problem) tracing the lineage of S objects implies essentially the detection of the guilty agents. Tutorial [4] provides a good overview on the research conducted in this field. Suggested solutions are domain specific, such as lineage tracing for data warehouses [5], and assume some prior knowledge on the way a data view is created out of data sources.

Our problem formulation with objects and sets is more general and simplifies lineage tracing, since we do not consider any data transformation from Ri sets to S. As far as 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 [16], video [8] and audio data [6] whose digital representation includes considerable redundancy. Recently, [1], [17], [10], [7] and other works have also studied marks insertion to relational data. Our approach and watermarking are 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. Finally, there are also lots of other works on mechanisms that allow only authorized users to access sensitive data through access control policies [9], [2]. 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.

IV. Agent Guilt Model

To compute this Pr{Gi|S}, we need an estimate for the probability that values in S can be "guessed" by the target. For instance, say some of the objects in S are emails of individuals. We can conduct an experiment and ask a person with approximately the expertise and resources of the target to find the email of say 100 individuals. If this person can find say 90 emails, then we can reasonably guess that the probability of finding one email is 0.9. On the other hand, if the objects in question are bank account numbers, the person may only discover say 20, leading to an estimate of 0.2. Probability pt is analogous to the probabilities used in designing fault-tolerant systems. That is, to estimate how likely it is that a system will be operational throughout a given period, we need the probabilities that individual components will or will not fail. A component failure in our case is the event that the target guesses an object of S. while we use the probability of guessing to identify agents that have leaked information.

The component failure probabilities are estimated based on experiments, just as we propose to estimate the pt's. Similarly, the component probabilities are usually conservative estimates, rather than exact numbers. For example, say we use a component failure probability that is higher than the actual probability, and we design our system to provide a desired high level of reliability. Then we will know that the actual system will have at least that level of reliability, but possibly higher. In the same way, if we use pt's that are higher than the true values, we will know that the agents will be guilty with at least the computed probabilities.

There are T={t1,t2,t3};R1={t1,t2};R2{t2,t3};S={t1,t2,t3}

In this case, all three of the distributor's objects have been leaked and appear in S. Let us first consider how the target may have obtained object t1, which was given to both agents. The target either guessed t1 or one of U1 or U2 leaked it. We know that the probability of the former event is p, so assuming that probability that each of the two agents leaked t1 is the same we have the following cases:

- The target guessed t1 with probability p;
- Agent U1 leaked t1 to S with probability (1 - p)/2;
- Agent U2 leaked t1 to S with probability (1 -p)/2;

Similarly, we find that agent U1 leaked t2 to S with Probability 1 - p since he is the only agent that has t2. Given these values, the probability that agent U1 is not Guilty, namely that U1 did not leak either object is:

$$(1 -(1 -p)/2) -(1 -(1 - p)); \qquad (1)$$

And the probability that U1 is guilty is: 1 – Pr {G1}

Note that if did not hold, our analysis would be more complex because we would need to consider joint events, e.g., the target guesses t1 and at the same time one or two agents leak the value. In our simplified analysis we say that an agent is not guilty when the object can be guessed, regardless of whether the agent leaked the value. Since we are "not counting" instances when an agent leaks information, the Simplified analysis yields conservative values (smaller Probabilities).

To simplify the formulas that we present in the rest of the paper, we assume that all T objects have the same pt, which we call p. Our equations can be easily generalized to diverse pt's though they become cumbersome to display. Next, we make two assumptions regarding the relationship among the various leakage events. The first assumption simply states that an agent's decision to leak an object is not related to other objects. In [14] we study a scenario where the actions for different objects are related, and we study how our results are impacted by the different independence assumptions.

Assumption 1 :For all t,t' Є 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 sources of a value t that appears in the leaked set. The Source can be any of the agents who have t in their sets or the target itself (guessing). To simply our formulas, the following assumption states that join events have a negligible probability. As we argue in the example below, this assumption gives us more conservative estimates for the guilt of agents, which is consistent with our goals.

Assumption 2: An Object t Є S can only be obtained by the target in one of the two ways as follows:

- ➔ A single agent U_i leaked t from its own R_i set.
- ➔ The target guessed (or obtained through other means) t without the help of any of the n agents.

In other words, for all t Є S, the event that the target guesses t and the events that agents $U_i(i=1\ldots\ldots\ldots n)$ leaks objects t are disjoint. Before we present the general formula for computing the probability Pr{Gi|S} that an agent Ui is guilty, we provide a simple example. Assume that the distributor set T, the agent sets R's and the target set S are:

T = {t1, t2, t3}, R1 = {t1, t2}, R2 = {t1, t3}, S = {t1, t2, t3}.

In this case, all three of the distributor's objects have been leaked and appear in S. Let us first consider how the target may have obtained object t1, which was given to both agents. From Assumption 2, the target either guessed t1 or one of U1 or U2 leaked it. We know that the probability of the former event is p, so assuming that probability that each of the two agents leaked t1 is the same we have the following cases:

- ➔ The target guessed t1 with probability p;
- ➔ Agent U1 leaked t1 to S with probability (1 . p)/2;
- ➔ Agent U2 leaked t1 to S with probability (1 . p)/2;

Similarly, we find that agent U1 leaked t2 to S with probability 1. p since he is the only agent that has t2. Given values, the probability U1 is guilty is:

$$\Pr\{G1|S\} = 1 - \Pr\{\bar{G}1\}. \qquad (2)$$

Note that if Assumption 2 did not hold, our analysis would be more complex because we would need to consider joint events, e.g., the target guesses t1 and at the same time one or two agents leak the value. Since we are "not counting" instances when an agent leaks information, the simplified analysis yields conservative values (smaller probabilities). In the general case (with our assumptions), to find the probability that an agent Ui is guilty given a set S, first we compute the probability that he leaks a single object t to S. To compute this we define the set of agents

Global Journal of Computer Science and Technology (C) Volume XII Issue XIII Version I Year 2012

$Vt = \{Ui|t \in Ri\}$ that have t in their data sets. Then using Assumption 2 and known probability p, we have:

$$Pr\{\text{some agent leaked t to S}\} = 1-p. \quad (3)$$

Assuming that all agents that belong to Vt can leak t to S with equal probability and using Assumption 2 we obtain:

$$Pr\{Ui \text{ leaked t to S}\} = \{\tfrac{1-P}{|Vt|}, \text{ if } Ui \in Vt, 0, \text{otherwise} \quad (4)$$

Given that agent Ui is guilty if he leaks at least one value to S, with Assumption 1 and Equation 4 we can compute the probability Pr {Gi|S} that agent Ui is guilty:

$$Pr\{Gi|S\} = 1 - \prod_{t \in S \cap Ri} \left(1 - \tfrac{1-P}{|Vt|}\right) \quad (5)$$

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. 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.

V. Future Enhancements

In this paper we are using multiple agents, for the purpose of security at the same time we are creating database for separate user, so through this we are strictly find out who is leaked information in internet.

References Références Referencias

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 for Composing 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: A Characterization 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 Data Warehouse Transformations,” The VLDB J., vol. 12, pp. 41-58, 2003.
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. Dependable and Secure Computing, vol. 2, no. 1, pp. 34-45, Jan.-Mar. 2005.
11. B. Mungamuru and H. Garcia-Molina, “Privacy, Preservation and Performance: The 3 P’s of Distributed Data Management, ”technical report, Stanford Univ., 2008.
12. V.N. Murty, “Counting the Integer Solutions of a Linear Equation with Unit Coefficients,” Math. Magazine, vol. 54, no. 2, pp. 79-81, 1981.
13. S.U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani, “Towards Robustness in Query Auditing,” Proc. 32nd Int’l Conf. Very Large Data Bases (VLDB ’06), VLDB Endowment, pp. 151-162, 2006.
14. P. Papadimitriou and H. Garcia-Molina, “Data Leakage Detection,” technical report, Stanford Univ., 2008.
15. P.M. Pardalos and S.A. Vavasis, “Quadratic Programming with One Negative Eigen value Is NP-Hard,” J. Global Optimization, vol. 1, no. 1, pp. 15-22, 1991.
16. J.J.K.O. Ruanaidh, W.J. Dowling, and F.M. Boland, “Watermarking Digital Images for Copyright Protection,” IEE Proc. Vision, Signal and Image Processing, vol. 143, no. 4, pp. 250-256, 1996.
17. R. Sion, M. Atallah, and S. Prabhakar, “Rights Protection for Relational Data,” Proc. ACM SIGMOD, pp. 98-109, 2003.
18. L. Sweeney, “Achieving K-Anonymity Privacy Protection Using Generalization and Suppression,” http://en.scientificcommons.