European Journal of Computer Science and Information Technology

Vol. 1 No. 1, pp. 1-10, June 2013

Published By European Centre for Research Training and Development UK (www.ea-journals.org)

DATA LEAKAGE DETECTION

Ms.Bangar Anjali N.
Ms.Rokade Geetanjali P.
Ms.Patil Shivlila
Ms.Shetkar Swati R.
Prof.N B Kadu
Pravara Rural Engineering College, Loni, Tal: Rahata, Dist: A, Nagar, Pin: 41373

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 found in an unauthorized place. 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

INTRODUCTION

Data leakage is the unauthorized transmission of data or information from within an organization to an external destination or recipient. Data leakage is defined as the accidental or intentional distribution of private or sensitive data to an unauthorized entity. Sensitive data of companies and organization includes intellectual property, financial information, patient information, personal credit card data and other information depending upon the business and the industry. A data distributor has given this sensitive data to a set of supposedly trusted agents (third parties). Some of the data are leaked and found in an unauthorized place. 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 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 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.

PROBLEM SETUP AND NOTATION

Entities and Agents

Let the distributor database owns a set S= {t1, t2… tm} which consists of data objects. Let the no of agents be A1, A2... An [6][10]. The distributor distributes a set of records S to any agents based on their request such as sample or explicit request.
• Sample request RI= SAMPLE (T, mi): Any subset of mi records from T can be given to Ui [1].
• Explicit request Ri= EXPLICIT (T;condi): Agent Ui receives all T objects that satisfy condition.
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. After giving objects to agents, the distributor discovers that a set S of 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 (A1, A2, ..., An) have 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.

Guilty Agents

Guilty agents are the agents who had leaked the data. Suppose the agent say Ai had leaked the data knowingly or unknowingly. Then automatically notification will be the send to the distributor defining that agent Ai had leaked the particular set of records which also specifies sensitive or non sensitive records. Our goal is to estimate the likelihood that the leaked data came from the agents as opposed to other sources.

Data Allocation Problem

The main focus of this paper is the data allocation problem: how can the distributor intelligently give data to agents in order to improve the chances of detecting a guilty agent. There are four instances of this problem, depending on the type of data requests made by agents and whether “fake objects” are allowed. Agent makes two types of requests, called sample and explicit.

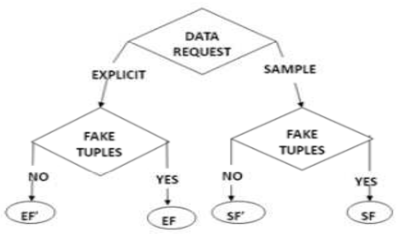


Fig.1 Leakage problem instances

Fake Objects

Fake objects are objects generated by the distributor that are not in set S. The objects are designed to look like real objects, and are distributed to agents together with the S objects, in order to increase the chances of detecting agents that leak data.

RELATED WORKS

The guilt detection approach we present is related to the data provenance problem: tracing the lineage of an S object implies essentially the detection of the guilty agents. It provides a good overview on the research conducted in this field. 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. 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, video and audio data whose digital representation includes considerable redundancy. 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. 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.

RESULTS OF DATA LEAKAGE DETECTION MODEL

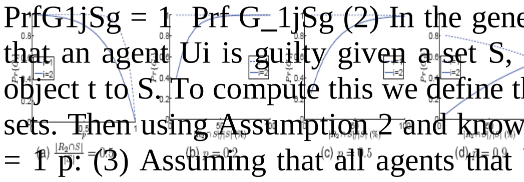
Agent Guilt Model

To compute this PrfGijSg, we need an estimate for the probability that values in S can be "guessed" bythe target. For instance, say some of the objects in T 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. We call this estimate pt, the probability that object t can be guessed by the target. 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.

Assumption 1. For all t; t0 2 S such that t 6= t0 the provenance of t is independent of the provenance of t0. To simplify our formulas, the following assumption states that joint 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 2 S can only be obtained by the target in one of two ways: A single agent Ui leaked t from his own Ri set; or The target guessed (or obtained through other means) t without the help of any of the n agents. In other words, for all t 2 S, the event that the target

guesses t and the events that agent Ui (i = 1; : : : ; n) leaks object t are disjoint. Before we present the general formula for computing PrfGijSg, we provide a simple example. Assume that sets T, R's and S are as
Follows: T = ft1; t2; t3g; R1 = ft1; t2g; R2 = ft1; t3g; S = ft1; t2; t3g:…..(Eqn 1) In this case, all three of the distributor's objectshave 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 the probability that each of the two agents leaked t1 is the same we have the following cases: the leaker 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 it is the only agent that has this data object. Given these values, the probability that agent U1 is not guilty, namely that U1 did not leak either object is: PrfG_1jSg = (1 (1 p)=2) _ (1 (1 p)) (1) Hence, the probability that U1 is guilty is:
PrfG1jSg = 1 Prf G_1jSg (2) 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 Vt = fUijt 2 Rig that have t in their data sets. Then using Assumption 2 and known probability p, we have: Presume agent leaked t to Sg = 1 p: (3) Assuming that all agents that belong to Vt can leak t to Swith equal probability and using Assumption 2 we obtain:……..(Eqn 2) 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 PrfGijSg that agent Ui is guilty:……..(Eqn 3)



Guilt Model Analysis
In order to see how our model parameters interact and to check if the interactions match our intuition, in this section, we study two simple scenarios. In each scenario, we have a target that has obtained all the distributor's objects, i.e., T ¼ S.

Impact of Probability p
In our first scenario, T contains 16 objects: all of them are given to agent U1 and only eight are given to a second agent U2. We calculate the probabilities PrfG1jSg and PrfG2jSg for p in the range [0, 1] and we present the results in Fig. 1a. The dashed line shows PrfG1jSg and the solid line shows PrfG2jSg. As p approaches 0, it becomes more and more unlikely that the target guessed all 16 values. Each agent has enough of the leaked data that its individual guilt approaches 1. However, as p increases in value, the probability that U2 isguilty decreases significantly: all of U2's eight objects were also given to U1, so it gets harder to blame U2 for the leaks.

On the other hand, U2's probability of guilt remains close to 1 as p increases, since U1 has eight objects not seen by the other agent. At the extreme, as p approaches 1, it is very possible that the target guessed all 16 values, so the agent's probability of guilt goes to 0. 5.2 Impact of Overlap between Ri and S In this section, we again study two agents, one receiving all the T ¼ S data and the second one receiving a varying fraction of the data. Fig. 1b shows the probability of guilt for both agents, as a function of the fraction of the objects owned by U2, i.e., as a function of jR2 \ Sj=jSj. In this case, p has a low value of 0.2, and U1 continues to have all 16S objects. Note that in our previous scenario, U2 has 50 percent of the S objects. We see that when objects are rare (p ¼ 0:2), it does not take many leaked objects before we can say that U2 is guilty with high confidence. This result matches our intuition: an agent that owns even a small number of incriminating objects is clearly suspicious. Figs. 1c and 1d show the same scenario, except for values of p equal to 0.5 and 0.9. We see clearly that the rate of increase of the guilt probability decreases as p increases. This observation again matches our intuition: As the objects become easier to guess, it takes more and more evidence of leakage (more leaked objects owned by U2) before we can have high confidence that U2 is guilty. In [14], we study an additional scenario that shows how the sharing of S objects by agents affects the probabilities that they are guilty. The scenario conclusion matches our intuition: with more agents holding the replicated leaked data, it is harder to lay the blame on any one agent.

EXISTING SYSTEM

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 agent. In some cases it is important not to alter the original distributor’s data. 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. *E.g.* 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.

PROPOSED SYSTEM

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 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.

Watermarking: Embedding & Extraction

A watermark in the insignificant part has helped to maintain the fidelity of the cover image. As seen from the results, imperceptibility is well preserved. Large capacity of watermarking is an added advantage of this scheme.Thus, large capacity watermark may be successfully embedded and extracted using this scheme, which can beextremely useful for companies engaged in developing watermarking applications and digital information security products. Embedding and extraction algorithms are used in this technique.

Steganography

Steganography is a technique for hiding a secret message within a larger one in such a way that others can't discern the presence or contents of the hidden message. A plain text message may be hidden in one of two ways ,the method of steganography conceal the existence of the message, whereas the outsiders of cryptography render the message unintelligible to transformation of the text.steganography serves as a means for private, secure and sometimes malicious communication.

DATA LEAKAGE DETECTION SYSTEM

Data Allocation Module

The main focus of our project is the data allocation problem as how can the distributor intelligently give data to agents in order to improve the chances of detecting a guilty agent, Admin can send the files to the authenticated user, users can edit their account details etc. Agent views the secret key details through mail. In order to increase the chances of detecting agents that leak data.

Fake Object Module

The distributor creates and adds fake objects to the data that he distributes to agents. Fake objects are objects generated by the distributor in order to increase the chances of detecting agents that leak data. The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents.

Optimization Module

The Optimization Module is the distributor's data allocation to agents has one constraint and one objective. The agent's constraint is to satisfy distributor's requests, by providing them with the number of objects they request or with all available objects that satisfy their conditions. His objective is to be able to detect an agent who leaks any portion of his data. User can able to lock and unlock the files for secure.

Data Distributor Module

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. 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.

Agent Guilt Module

Probability of guilt Pr {Gi|S} can be computed by estimating the probability that the target can guess objects in “S”. The proposed guilt model makes two assumptions. The first assumption is that the source of a leaked object can be of any agent. The second assumption is that An object which is part of set of objects distributed can only be obtained from one of the agents or through other means. With these assumptions the probability of guilt is computed as Pr{Ui leaked t to S}
= { 1-p , if Ui∈Vt
|Vt|
0, otherwise

DATA ALLOCATION STRATEGIES

The data allocation strategies used to solve the problem of data distribution as discussed in previous sections exactly or approximately are provided in the form of various algorithms. The algorithms are provided here.

Explicit Data Request

```
Algorithm 1 Allocation for Explicit Data Requests (EF)

Input: $R_1, \ldots, R_n$, $cond_1, \ldots, cond_n$, $b_1, \ldots, b_n$, B
Output: $R_1, \ldots, R_n$, $F_1, \ldots, F_n$
1: R ← ϕ                ▹ Agents that can receive fake objects
2: for i=1, .......,n do
3: if $b_i > 0$ then
4: R ← R U {i}
5: $F_i$ ← ϕ
6: while B> 0 do
7: i ← SELECTAGENT(R, $R_1, \ldots, R_n$)
8: f ← CREATEFAKEOBJECT ($R_i$, $F_i$, $cond_i$)
9: $R_i$ ← $R_i$ U {f}
10: $F_i$ ← $F_i$ U {f}
11: $b_i$ ← $b_i$ -1
12: if $b_i$ = 0 then
13: R ← R\{$R_i$}
14: B← B-1
```

Fig.2 – Allocation for explicit data requests

It is a general algorithm that is used by other algorithms.

```
Algorithm 2 Agent Selection for e-random
1: function SELECTAGENT(R, $R_1, \ldots, R_n$)
2: i← select at random an agent from R
3: return i
```

Fig.3 – Agent selection for e-random

This algorithm actually performs random selection of objects.

```
Algorithm 3 Agent selection for e-optimal

1: function SELECTAGENT (R,R_1,.....,R_n)
2: i ← $\underset{i':R_{i'} \in R}{\operatorname{argmax}} \left(\frac{1}{|R_{i'}|} - \frac{1}{|R_{i'}|+1}\right) \Sigma_j |R_{i'} \cap R_j|$
3: return i
```

Fig. 4 – Agent selection for e-optional

This algorithm actually performs random selection of objects.

```
Algorithm 3 Agent selection for e-optimal

1: function SELECTAGENT (R,R_1,.....,R_n)
2: i ← $\underset{i':R_{i'} \in R}{\operatorname{argmax}} \left(\frac{1}{|R_{i'}|} - \frac{1}{|R_{i'}|+1}\right) \Sigma_j |R_{i'} \cap R_j|$
3: return i
```

Fig.5 – Agent selection for e-optional.

Sample Data Request

This algorithm is meant for making a greedy choice of choosing an agent that causes improvement in the sum-objective.

```
Algorithm 4: Allocation for Sample Data Requests(SF)

Input: $m_1$,..........,$m_n$, |T|          ▷ Assuming $m_i \le |T|$
Output: $R_1$,......$R_n$
1: a ← $0_{|T|}$          ▷ a[k]: number of agents who have received object $t_k$
2: $R_1$ ← ϕ,........., $R_n$ ← ϕ
3: remaining ← $\Sigma^n_{i=1} m_i$
4: while remaining > 0 do
5: for all i=1,.......,n:|$R_i$| < $m_i$ do
6: k ← SELECTOBJECT (i, $R_i$)   ▷ May also use additional parameters
7: $R_i$ ← $R_i$ U {$t_k$}
8: a[k] ← a[k] + 1
9: remaining ← remaining-1
```

Fig. 6 – Allocation for sample data requests

AES Algorithm

The AES algorithm is based on permutations and substitutions. Permutations are rearrangements of data, and substitutions replace one unit of data with another. AES performs permutations and substitutions using several different techniques. The four operations SubBytes, ShiftRows, MixColumns, and AddRoundKey are called inside a loop that executes Nr times—the number of rounds for a given key size, less 1. The number of rounds that the encryption algorithm uses is 10, 12, or 14 and depends on whether the seed key size is 128, 192, or 256 bits. In this example, because Nr equals 12, the four operations are called 11 times. After this iteration completes, the encryption algorithm finishes by calling SubBytes, ShiftRows, and AddRoundKey before copying the State matrix to the output parameter. In summary, there are four operations that are at the heart of the AES encryption algorithm. AddRoundKey substitutes groups of 4 bytes using round keys generated from the seed key value. SubBytes substitutes individual bytes using a substitution table. ShiftRows permutes groups of 4 bytes by rotating 4-byte rows. MixColumns substitutes bytes using a combination of both field addition and multiplication.

CONCLUSION

From this study we conclude that the data leakage detection system model is very useful as compare to the existing watermarking model. We can provide security to our data during its distribution or transmission and even we can detect if that gets leaked. Thus, using this model security as well as tracking system is developed. Watermarking can just provide security using various algorithms through encryption, whereas this model provides security plus detection technique.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.

ACKNOWLEDGEMENT

For all the efforts behind the paper work, we first & foremost would like to express our sincere appreciation to the staff of Dept. of Computer Engg., for their extended help & suggestions at every stage of this paper. It is with a great sense of gratitude that we acknowledge the support, time to time suggestions and highly indebted to our guide **Prof. N. B. Kadu and** Prof**. S. D. Jondhale (H.O.D.).** Finally, we pay sincere thanks to all those who indirectly and directly helped us towards the successful completion of the paper.

REFERANCES

[1] Sandip A.Kale, Prof. Kulkarni S.V. (Department Of Computer Sci. &Engg,MIT College of Engg, Dr.B.A.M.University, Aurangabad(M.S), India, Data Leakage Detection: A Survey, (IOSR Journal of Computer Engineering (IOSRJCE)ISSN : 2278-0661 Volume 1, Issue 6 (July-Aug 2012), PP 32-35 www.iosrjournals.org.

[2]R. Agrawal and J. Kiernan, “Watermarking Relational Databases,”Proc. 28th Int’l Conf. Very Large Data Bases (VLDB ’02), VLDB Endowment, pp. 155-166, 2006

[3] IEEE Transactions On Knowledge And Data Engineering, Vol. 22,No. 3, March 2011 Data Leakage Detection Panagiotis Papadimitriou, Member, IEEE, Hector Garcia-Molina, Member, IEEE P.P (2,4-5)

[4] IOSR Journal of Computer Engineering (IOSRJCE) ISSN: 2278-0661, ISBN: 2278-8727 Volume 5,Data Allocation strategies for Leakage Detection Issue 2 (Sep-Oct. 2012), PP 30-35 www.iosrjournals.org

[5] International Journal of Advanced Research in Computer and Communication Engineering Vol. 1, Data Leakage Detection Issue 9, November 2012..

[6] International Journal of Communications and Engineering Volume 04– No.4, Issue: 03 March2012 Data Leakage Detection And Prevention Using Perturbation And Unobtrusive Analyze