IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661, ISBN: 2278-8727 Volume 5, Issue 2 (Sep-Oct. 2012), PP 30-35
www.iosrjournals.org

Data Allocation Strategies for Leakage Detection

Sridhar Gade[1], Kiran Kumar Munde[2], Krishnaiah.R.V.[3]
[1]*Department of CSE, DRK Institute of Science & Technology, Ranga Reddy, Andhra Pradesh, India*
[2]*Department of CSE, DRK College of Engineering & Technology,Ranga Reddy, Andhra Pradesh, India*
[3]*Principal Department of CSE, DRK Institute of Science & Technology,Ranga Reddy, Andhra Pradesh, India*

Abstract: *Data plays a pivotal role in IT systems. Especially when sensitive data has to be sent to other places through trusted agents, it is very challenging and important to detect leakage when they deliberately leak it to others. The scenario where a distributor gives sensitive data to his trusted agents and the data is intentionally leaked to others. The distributor should identify or detect this leakage and its means that is who leaked it as well. This is the problem this paper intended to solve. Towards this we propose new data allocation strategies for improving the probability of detecting leakages accurately. The system should detect leakage correctly and the means as well as against to the leakage by other means. The proposed methods do not relay on the alterations of released data. It is also possible to inject "looks genuine but fake" data in order to improve the probability of detecting leakage and tracing the party who actually leaked it.*

Index Terms *– Data leakage, leakage detection model, data allocation strategies, fake records*

I. Introduction

In business applications data can be transmitted securely through network. Due the emergence of many cryptographic algorithms, end to end security methods, it is possible to send data across the machines with full security. However, there is possibility for online attacks. The security in this case depends on the strength of cryptographic algorithms. This is one side of the coin. The other side of the coin is that in business scenarios people need to send information through trusted parties. In this case the distributor of data is fully aware that the data leakage may happen. However, the distributor has trust over the agents who carry his data to other destinations to which the distributed is associated for business purposes. Provided this scenario, the distributor can only hope genuine behavior from his trusted agents. What if the agents behave quite opposite to the belief of distributor? is the important question answered by this paper. When data is leaked by trusted agents, there should be some way to identify it and prove it. Unfortunately this is the job difficult to achieve. Other scenarios where data has to be distributed through trusted agents include patients records may be given by hospital; sharing of data is required among companies with partnerships; an enterprise may decide to outsource it data process and hence need to handover the valuable data to other party. In all these scenarios, the provider of sensitive data is considered as distributor.

The aim of this paper is to detect leakage no matter who is involved in leakage and proving that data has been leaked. One naïve technique is to modify and make it "less sensitive" before actually giving to trusted agents. The alterations may be done by introducing noise in the data or replace certain values and remember them [1]. But it is not good practice to modify original data. To ensure this data leakage detection is done using watermarking traditionally. A unique symbol is embedded into each copy that has been distributed. When such symbol is found with any unauthorized person, it is the proof that data leakage has been occurred. Watermarking is effective in leakage detection. However, it involves modification of original data. There is security problem with this. When receiver is malicious, it can be destroyed. This paper proposes a novel technique that ensures that data leakage is detected without actually modifying data. It is achieved like this. When data is given trusted agents and found that leaked to other parties who are not authorized, the proposed system can identify the leakage and also identify the means of leakage. The distributor can find out the likelihood of data leakage and means of leakage. This is achieved by using algorithms for distributing objects to trusted agents in such a way that it improves possibility of data leakage detection. The algorithms also consider adding "fake" objects to the set of distributed objects in order to improve the possibilities of detecting leakage. The fake objects are not at all related to real objects but appear so in the eyes of agents. The fake objects are indirectly acting as watermark in this case. When any agent finds fake objects in somewhere, he can suspect that particular agent to be guilty of leakage.

II. Problem Statement

We take a hypothetical problem in which a distributor owns a set of objects. He wants to share those objects with a group of humans known as trusted agents. The distributor does not want the objects shared with agents to be leaked to third parties. The objects may be of any type of any size. They could be records in relational tables or files in file system. Agents get some or all of the objects based on the requirement. The trusted agents are believed to be trusted. However, when they involve in any fraud activities the data gets leaked. This is the problem addressed in this paper. Towards this a guilt model is proposed. After receiving objects from distributor the trusted agents may misbehave and leak the data objects to some third party. When the leaked objects are viewed by distributor through any means, he can suspect that those objects are given by one of his trusted agents. The aim of this paper is to prove that the data is leaked by so and so agent by proposing data allocation strategies.

III. Related Work

Data leakage detection has been around with respect to IT systems. Security threats like impersonation, hacking, intrusion, eves dropping and VIRUS can be prevented using security software available. All forms of electronic exchange of data have security mechanisms in place. However, guilt detection in a scenario where data is handed over to trusted agents (humans) and expect them to transfer data to intended recipients is a challenging task. The following review establishes facts in line with this problem. Data provenance problem has been around and it is related to data origin and the originality of data. In [2] a data provenance problem is discussed which is relevant to the guilt detection problem presented in this paper. By tracing the origin of given objects does mean that tracing the probability of guilt. Further research in this field is presented in a tutorial [3] which reviews all possible causes and probability of proving data provenance problem. The solutions in this area are domain specific and they are pertaining to data warehouses [4] assuming to have prior know how on data sources and the way data is created. In this paper, out problem formulation is simple and general and does not alter the original objects to be distributed. When a set of objects are to be distributed thorough trusted agents, we formulate objects that are not changed as opposed to watermarking. Lineage tracing is performed without using watermarking here. Watermarking technology has been around to protect intellectual property of people that is in electronic format. However, it needs the object that needs to be protected to be modified in order to embed some sort of watermark for security reasons. When watermarked image is tampered that is made well known to the distributor thus establishing the fraud taken place. Watermarking can be used with images [4], audio [5] and video [6]. These media's digital data has redundancy. Relational data can also be protected using something similar to watermarking. This is achieved by inserting some marks into the data for security reasons. This kind of research is reviewed in [7], [8], and [9].

Our approach in this paper and watermarking are similar in the sense of providing identification of information for originality. However, they are totally different as our approach does not need to alter objects to be distributed as opposed to watermarking. There are other research works that focused on enabling IT systems to ensure that only intended receivers will receive data. It is achieved access control policies proposed in [10] and [11]. These policies help in protecting data when it is transferred and detect leakage of data as well. However, they are very restrictive in nature and it is impossible for them to satisfy requests from agents.

IV. Agent Guilt Model

Probability of guilt Pr {Gi|S} can be computed by estimating the probability that the a 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

$$P_r\{U_i \text{ leaked } t \text{ to } S\} = \begin{cases} \dfrac{1-p}{|V_t|}, & \text{if } U_i \in V_t \\ 0, & \text{otherwise} \end{cases}$$

V. Analysis Of Guilt Model

In this section our guilt modeling is analyzed to see whether it works correctly. Two simple scenarios we take and in each case all distributed objects are obtained by target i.e., T=S. Assuming that T has 16 objects. Out of them only 8 are given to U2 and all of them are given to U1. Probability of guild for both the users and agents is calculated. The results are as given in fig. 1, 2, 3 and 4.

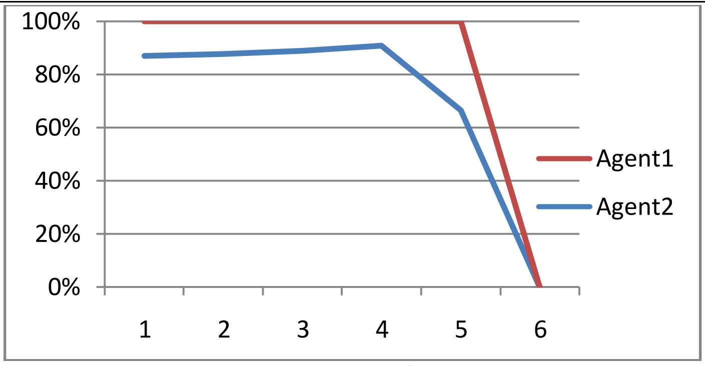

Fig. 1 – Guilt probability as a function with p = 0.5

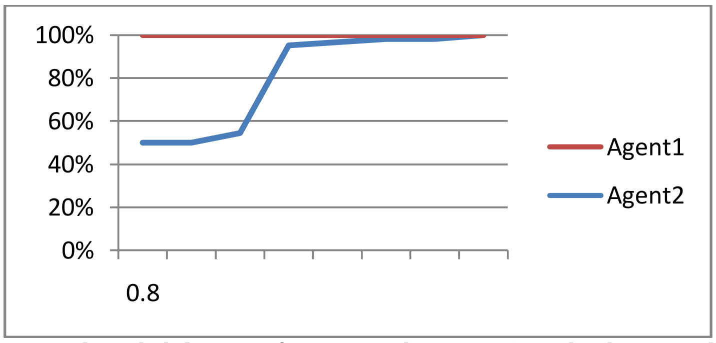

Fig. 2 – Guilt probability as a function with p = 0.2 (Overlap b/w S and R2)

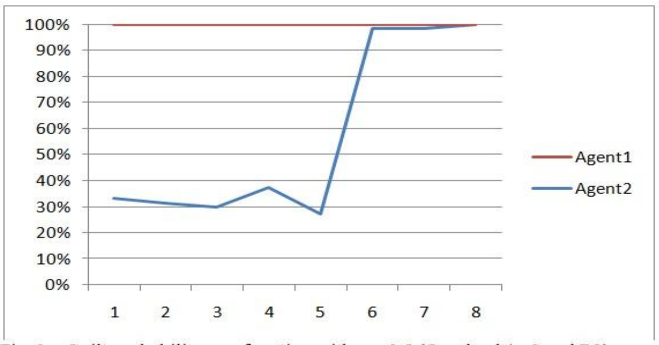

Fig. 3 – Guilt probability as a function with p = 0.5 (Overlap b/w S and R2)

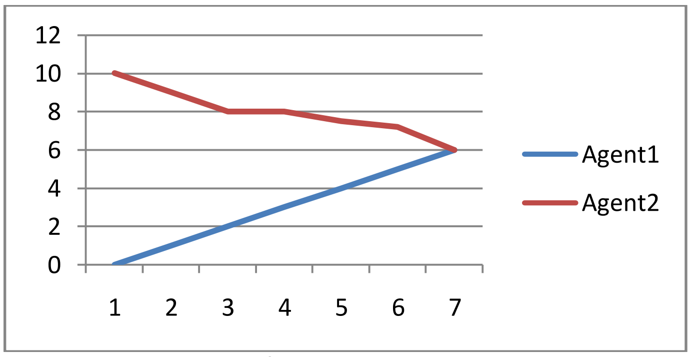

Fig. 4 – Guilt probability as a function with p = 0.9 (Overlap b/w S and R2)

As can be seen in above figures when p value is 0 it is not likely that all 16 objects are guessed by target. Each agent has some leaked and approaches 1. The probability that U2 is guilty decreases when p value increases. However, the probability of U2's guilt remains more and close to 1 as agent 2 has 8 values that are not known to other agent. When p value approaches 1, the agent's probability of guilt becomes zero.

VI. Data Allocation Problem Description

Data allocation is the main focus in this paper. Distributor is supposed to allocate data objects to trusted agents intelligently. Two types of requests are handled namely sample and explicit. While distributing objects, fake objects that mimic real objects are created and given to agents along with real objects. The fake objects are created in such a way that the agent's information who carries the objects is kept in those objects. The intention of creating fake objects is to maximize the probabilities of detecting guilt agents. The fake objects created are given to trusted agents along with the actual and real objects. The fake objects are somehow associated with information of agent who carries them. The whole thing is transparent to agents as they can't distinguish between the fake and real objects. The process of creating fake objects has to be done carefully and intelligently. While creating fake objects, the distributor can also specify certain limit for fake objects so as to ensure that the agents do not suspect some of the objects as fake. The fake objects look like real objects and the agents have no knowledge as to how to distinguish between fake and real objects.

In order to optimize the data allocation process a distributor must has a constraint and also an objective. The constraint is that distributor has to send objects required by agents. Objective of distributor is having the ability to detect an agent when objects are leaked. The distributor's objective is calculated as
$\Delta(i,j) = P_r\{G_i|R_i\} - P_r\{G_j|R_i\}$ $i,j = 1,\ldots\ldots.,n$

VII. Data Allocation Techniques

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.

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

Input: $R_1,\ldots\ldots,R_n$, $cond_1,\ldots\ldots,cond_n$, $b_1,\ldots\ldots,b_n$, 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. 5 – 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, $R_1,\ldots\ldots,R_n$)
2: i← select at random an agent from R
3: return i
```

Fig. 6 – Agent selection for e-random

This algorithm actually performs random selection of objects.

```
Algorithm 3 Agent selection for e-optimal

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

Fig. 7 – Agent selection for e-optional

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

```
Algorithm4: Allocation for Sample Data Requests(SF)

Input: $m_1,\ldots\ldots\ldots,m_n$, |T|                ▹ Assuming $m_i \le |T|$
Output: $R_1,\ldots..,R_n$
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. 8 – Allocation for sample data requests

This is a general algorithm that is required by other algorithms.

```
Algorithm 5: Object Selection for s-random

1: function SELECTOBJECT (i,$R_i$)
2: k ← select at random an element from set {k` | $t_k \notin R_i$}
3: return k
```

Fig. 9 – Object selection for s-random

This algorithm is meant for random selection of objects.

```
Algorithm 6: Object Selection of s-overlap

1: function SELECTOBJECT (i, $R_i$, a)
2: K ← { k| k = argmin a[k`]}
3: k ← select at random an element from set {k` | k` □K $\wedge$ $t_{k`}$, $\notin R_i$}
4: return k
```

Fig. 10 – Object Selection of s-overlap

This algorithm is meant for selection of objects in s-overlap fashion.

```
Algorithm 7 Object Selection for s-max

1: function SELECTOBJECT (i, $R_1$,……,$R_n$,$m_1$,……,$m_n$)
2: min_overlap ← 1    ▹ the minimum out of the maximum relative overlaps that the allocations of different objects to $U_i$ yield
3: for k ∈ { k` | $t_{k`}$, $\notin R_i$ } do
4: max_rel_ov ← 0  ▹ the maximum relative overlap between $R_i$ and any set $R_j$ that the allocation of $t_k$ to $U_i$ yields
5: for all j=1,…..,n: j≠i and $t_k \in R_j$ do
6: abs_ov ← |$R_i \cap R_j$| + 1
7: rel_ov ← abs_ov/min($m_i$, $m_j$)
8: max_rel_ov ← MAX (max_rel_ov, rel_ov)
9: if max_rel_ov ≤min_overlap then
10: min_overlap ← max_rel_ov
11: ret_k ← k
12: return ret_k
```

Fig. 11 – Object selection for s-max

This algorithm defines a new SELECTOBJECT() procedure, used to select objects to ahcive minimum increase of maximum relative overlap among agents.

VIII. Emperical Results

The environment used for the experiments include Windows XP OS, Java programming language, Eclipse IDE. A prototype application has been built in order to simulate the data leakage detection process. The results showed in fig. 5 and 6 show the results with respect to e-optional, e-random and no fake algorithms.

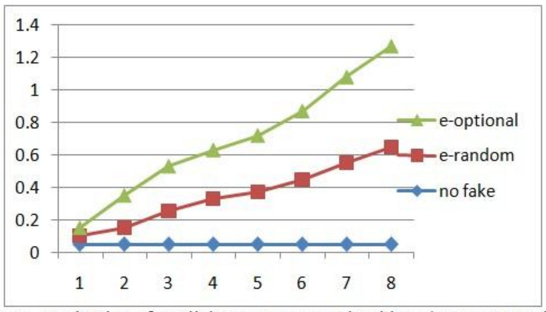

Fig. 5 – Evaluation of Explicit Data Request Algorithms (Average Metric)

Fig. 5 – Evaluation of Explicit Data Request Algorithms (Average Metric)

As can be seen in fig. 5, it shows average metric is affected by allocation of fake objects. The straight line in the graph represents that object allocation is done without fake objects.

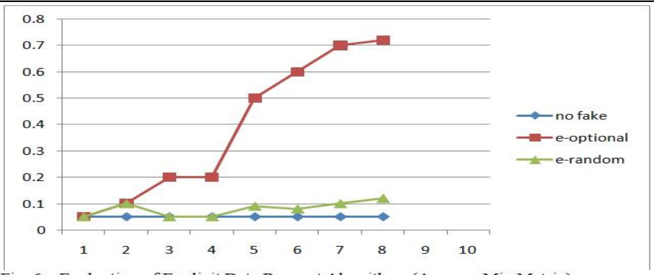

Fig. 6 – Evaluation of Explicit Data Request Algorithms (Average Min Metric)

As can be seen in fig. 6, it shows average metric is affected by allocation of fake objects. The straight line in the graph represents that object allocation is done without fake objects.

IX. Conclusion

When the world is not perfect in conduct and behavior and you need to send sensitive data to intended recipient through trusted agents, it is essential to have monitoring on the distribution process. When sensitive data has to be sent through electronic means, there are many security systems that can protect the data when it is on transit and also ensure that it reaches only to the intended recipient in original format. This paper addresses a different problem where data transmission takes place through human beings known as trusted agents. Detecting probability of data leakage has paramount importance especially when the data is confidential and sensitive in nature. We considered a scenario where a distributor is supposed to send sensitive data through his trusted agents and needs to detect when data is leaked by trusted agents in any fashion. The establishment of the probability of leakage and identifying the agent who leaked it is a challenging task. To address this problem, we proposed data allocation strategies that are personalized in such a way that when leaked data is found somewhere, it is possible to identify the agent who leaked it as agent's information is embedded somewhere as part of the strategies. Unlike watermarking which modifies original objects before being transmitted for security reasons, our system does not need any modification of original objects. Instead we introduce fake objects that are personalized in terms of agents who carry them. The fake objects are given to agents along with real objects that are transparent to trusted agents. When they leak the data for any reason and when distributor finds the leaked data, the proposed system helps the distributor to identify the agent who caused leakage. We implemented various algorithms that are having different data allocation strategies meant for enhancing the probabilities of distributor in identifying the leaker. In future we work on the agent guilt models that are not discussed in this paper and also enhance the distribution strategies further to make it more robust to data leakage.

References

[1] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression, 2002.

[2] 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.

[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] 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.

[5] S. Czerwinski, R. Fromm, and T. Hodes.Digital music distribution and audio watermarking.

[6] F. Hartung and B. Girod.Watermarking of uncompressed and compressed video.*Signal Processing*, 66(3):283–301, 1998.

[7] 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.

[8] F. Guo, J. Wang, Z. Zhang, X. Ye, and D. Li. *Information Security Applications*, pages 138–149.Springer, Berlin / Heidelberg, 2006. An Improved Algorithm to Watermark Numeric Relational Data

[9] Y. Li, V. Swarup, and S. Jajodia. Fingerprinting relational databases: Schemes and specialties.*IEEE Transactions on Dependable and Secure Computing*, 02(1):34–45, 2005.

[10] 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.

[11] S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Trans. Database Syst.*, 26(2):214–260, 2001.

	Sridhar Gade is a student of DRK Institute of science and Technology, Ranga Reddy, Andhra Pradesh, India. He has received M.Scdegree in Computer Science and M.Tech Degree in Computer Scienceand Engineering. His main research interest includes Data Mining ,Networking
	Kiran Kumar Mundeis a student of DRK College of Engineering & Technology, Ranga Reddy, Andhra Pradesh, India. He has received M.C.A andM.Tech Degree in Computer Scienceand Engineering. His main research interest includes Data Mining ,Software Engineering.
	Dr.R.V.Krishnaiah(Ph.D) is working as Principal at DRK INSTITUTE OF SCINCE & TECHNOLOGY, Hyderabad, AP, INDIA. He has received M.TechDegree(EIE&CSE). His main research interest includes Data Mining, Software Engineering.