Vol. 2, Issue 3, May-Jun 2012, pp. 404-410

Data leakage Detection

Rohit Pol (Student), Vishwajeet Thakur (Student), Ruturaj Bhise (Student), Guide –Prof. Akash Kate

Maharashtra Academy of Engineering, University of Pune,
Pune, Maharashtra, India

Abstract

In both the commercial and defence sectors a compelling need is emerging for rapid, yet secure, dissemination of Information. In this paper we address the threat of information leakage that often accompanies such information flows. We focus on domains with one information source (sender) and many information sinks (recipients) where: (i) sharing is mutually beneficial for the sender and the recipients, (ii) leaking a shared information is beneficial to the recipients but undesirable to the sender, and (iii) information sharing decisions of the sender are determined using imperfect monitoring of the (un)intended information leakage by the recipients. We make two key contributions in this context: First, we formulate data leakage prevention problems as Partially Observable Markov Decision Processes; we show how to encode one sample monitoring mechanism—digital watermarking—into our model. Second, we derive optimal information sharing strategies for the sender and optimal information leakage strategies for a rational-malicious recipient as a function of the efficacy of the monitoring mechanism. We believe that our approach offers a first of a kind solution for addressing complex information sharing problems under uncertainty.

1. Introduction

We handle 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 (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 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.

1.1 Problem Statement

Identifying data leakages and improve the probability. 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.

1.2 Objectives

- 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 (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 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.
- 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.
-

Vol. 2, Issue 3, May-Jun 2012, pp. 404-410

2. Literature Survey

2.1 Existing System

In existing system, 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. However, 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.

2.2 Proposed System

In proposed system, 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. In this project 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. 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.

2.3 Steganography

2.3.1 About

The recently created technology of steganography entices a wide audience from the casual observer, to the scientific professional. The various applications of steganography must be scrutinized in order to understand the future progression of this technology. This paper attempts to reveal new and current angles of steganography. What are revealed in this essay are the ingenious history and background of steganography in examples and its interwoven development path with cryptography, even to this day. On a technical level this paper tests the steganography community's theory that, in general, the stego process diminishes contrast within a digital photo.

Steganography is obscure in its recurrent appearance and disappearance throughout history. On a sociological level, having finally taken a foothold during the internet generation's explosive growth of the 1990s, steganography serves as a means for private, secure and sometimes malicious communication.

2.3.2 Inference

The recently created technology of steganography entices a wide audience from the casual observer, to the scientific professional. The various applications of steganography must be scrutinized in order to understand the future progression of this technology. This paper attempts to reveal new and current angles of steganography. What are revealed in this essay are the ingenious history and background of steganography in examples and its interwoven development path with cryptography, even to this day. On a technical level this paper tests the steganography community's theory that, in general, the stego process diminishes contrast within a digital photo.

Steganography is obscure in its recurrent appearance and disappearance throughout history. On a sociological level, having finally taken a foothold during the internet generation's explosive growth of the 1990s, steganography serves as a means for private, secure and sometimes malicious communication.

2.4 Data Leakage Detection

2.4.1 About

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 (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 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.

2.4.2 Inference

In a perfect world there would be no need to hand over sensitive data to agents that may unknowingly or maliciously leak it. And even if we had to hand over sensitive data, in a perfect world we could watermark each object so that we could trace its origins with absolute certainty. However, in many cases we must indeed work with agents that may not be 100% trusted, and we may not be certain if a leaked object came from an agent or from some other source, since certain data cannot admit watermarks.

In spite of these difficulties, we have shown it is possible to assess the likelihood that an agent is responsible for a leak, 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. Our model is relatively simple, but we believe 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.

3. System Requirements Specification

3.1 Aim of the project

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.

3.2 Description of the project in short

In this project we are finding out the data is leaked or not. The age will give the information to broadcast the data on the news media. We will check whether the authorized user leaked the data to another news media channel.

3.3 Process Summary

In this project authorized agent give the request for data request may be explicit or sample, according to request agent get the data.

If agent leaked the data into another channel then in our side we are checking whether the data is match with our website and the second website. After that we will find out the agent who leaked the data.

3.4 Algorithms

Allocation for Explicit Data Requests

In this request the agent will send the request with appropriate condition.

Agent gives the input as request with input as well as the condition for the request after processing the data after processing on the data the gives the data to agent by adding fake object with an encrypted format.

Allocation for Sample Data Requests

In this request agent request does not have condition. The agent sends the request without condition as per his query he will get the data.

3.5 Deliverables

1. Web site for the news media
2. Database backup
3. 1 Another website news media

3.6 Operating Environment

3.6.1 Hardware

- Processor : Intel (R) Core(TM) i3 CPU
- Installed RAM : 1 GB
- System type : 32 bit operating systems

3.6.2 Software

- Java 1.6
- My SQL.

3.7 Assumptions and Dependencies

Assumption 1

Each and every agent has its unique data.

Assumption 2

Another news channel will received the data only from agent.

3.8 Modules Information

3.8.1 Module1

In this module we will give the design for the website as well as the authorized agent will logged in. for the new agent the has to fill the register form, after successful submission the of data he can logged into our system.

3.8.2 Module2

In this module the agent will fired a query, he will get the data from database as per query.

3.8.3 Module3

This module gives agent data by adding the fake object; depend on the agent he will leak the data to another new channel. After leaked the data another news channel updated with that data.

3.8.4 Module4

This module checks whether our system data as well as another system data is same or not.

3.9 Project Plan

Plan of Execution

- Identification – Searching for different project ideas. Identifying and finalizing one of them for further implementation.
- Conceptualization and design – The concepts required for building the project are studied in detail. Also the high level designing is done at this stage. Preliminary presentation given for more clarification of project.
- Detailed design – At this stage, low level designing is done. User Interface is designed to give better visualization of the project idea.
- Coding – Actual implementation of the project starts at this stage. Coding for each module of the four modules will be done. Coding and testing will take approximately 8 to 10 weeks.
- Unit Testing –Initially the backend database will be tested over no of transactions. Then GUI for mobile user as well as for HTTP user is tested separately.
- Integration Testing –All modules will be integrated and then testing of whole integrated testing will be performed. It also includes evaluation of project.

- System Testing – The product was tested in the context of the entire system. Different Linux systems will be used for system testing and the performance will be monitored.
- Documentation – A detailed document about the project shall be prepared at this stage.

4. Requirement Analysis

4.1 Problem Definition

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 (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 having been independently gathered by other means. 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 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.

4.2 Project Objectives

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 (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 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.

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.

4.3 Software

- Java 1.6
- My SQL.

4.4 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 = S.

4.4.1 Impact of Probability (p)

In our first scenario, T contains 16 objects: all of them are given to agent U_1 and only 8 are given to a second agent U_2. We calculate the probabilities PrfG1jSg and

PrfG2jSg for p in the range [0, 1] and we present the results in Figure 1(a). 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 U_2 is guilty decreases significantly: all of U_2's 8 objects were also given to U_1, so it gets harder to blame U_2 for the leaks. On the other hand, U_2's probability of guilt remains close to 1 as p increases, since U_1 has 8 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.

4.4.2 Impact of Overlap between R_i and S

In this subsection we again study two agents, one receiving all the T = S data, and the second one receiving a varying fraction of the data. Figure 1(b) shows the probability of guilt for both agents, as a function of the fraction of the objects owned by U_2, i.e., as a function of j $R_2 \setminus S_j = S_j$. In this case, p has a low value of 0.2, and U_1 continues to have all 16 S objects. Note that in our previous scenario, U_2 has 50% of the S objects. We see that when objects are rare (p = 0.2), it does not take many leaked objects before we can say U_2 is guilty with high confidence. This result matches our intuition: an agent that owns even a small number of incriminating objects is clearly suspicious.

Figures 1(c) and 1(d) 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 U_2) before we can have high confidence that U_2 is guilty.

5. Conclusion and Future Scope

5.1 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.

5.2 FUTURE SCOPE

Our future work includes the investigation of agent guilt models that capture leakage scenarios that are not studied in this paper. For example, what is the appropriate model for cases where agents can collude and identify fake tuples?

Another open problem is the 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).

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.

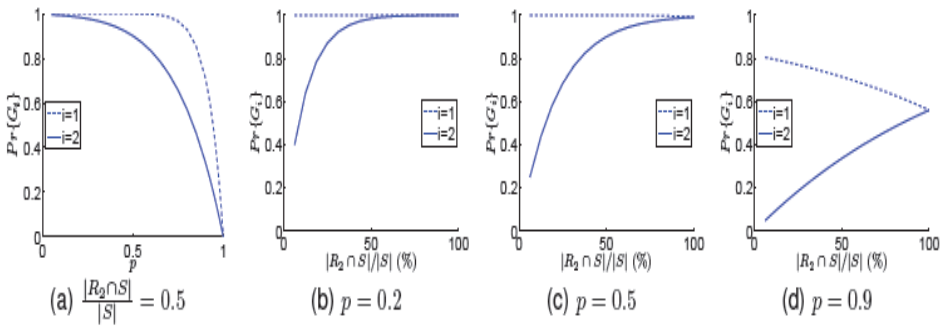
[4] P. Buneman and W.-C. Tan. Provenance in databases. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 1171–1173, New York, NY, USA, 2007. ACM.

[5] Y. Cui and J. Widom. Lineage tracing for general data warehouse transformations. In The VLDB Journal, pages 471–480, 2001.

Vol. 2, Issue 3, May-Jun 2012, pp. 404-410

Sr. No	Module For	**S/W Required for coding**	**Period for coding** ***(Approx. weeks)***	**S/W Required for Testing**	**Period for Testing** ***(Approx. weeks)***
1.	Requirement gathering	N/A	1	N/A	N/A
2.	Analysis	N/A	1	N/A	N/A
3.	Creating Application	Java	2	Jcreator	1
4.	Integrating all the modules	Java	1	Jcreator	N/A
5.	Final Testing	---------	1	N/A	N/A

Table 3.1 Plan of Execution



(a) $\frac{|R_2 \cap S|}{|S|} = 0.5$ (b) $p = 0.2$ (c) $p = 0.5$ (d) $p = 0.9$

Fig. 1. Guilt probability as a function of the guessing probability p (Figure (a)) and the overlap between S and R_2 (Figures (b), (c) and (d)). In all scenarios it holds that $R_1 \cap S = S$ and $|S| = 16$.