Efficient Detect File Watcher for Data Leakage

Doolla Sridhar, Kunapareddy Rajani Devi

1,2Dept. of IT, Nalanda Institute of Engineering & Tech. Sattenapalli, Guntur, AP, 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 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 File Watcher technique for data leakage and follows data allocation strategies (across the agents) that improve the probability of identifying leakages. The proposed technique should be able to work with common protocols and technologies. 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

Data Mining, 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 have 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 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 [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 leak 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 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 website, 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 we catch Freddie with a single cookie, he can argue that a friend gave him the cookie. But if we catch Freddie with five 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.

Data Loss Prevention (DLP) is a systems that identify, monitor, and protect data in use (e.g. endpoint actions), data in motion (e.g. network actions), and data at rest (e.g. data storage) through deep content inspection, contextual security analysis of transaction (attributes of originator, data object, medium, timing, recipient/destination and so on) and with a centralized management framework. Systems are designed to detect and prevent unauthorized use and transmission of confidential information. Vendors refer to the term as Data Leak Prevention, Information Leak Detection and Prevention (ILDP), Information Leak Prevention (ILP), Content Monitoring and Filtering (CMF), Information Protection and Control (IPC) or Extrusion Prevention System by analogy to Intrusion-prevention system.

In this paper, we are introducing a Data leak file watcher technique over the common protocols and technologies and the system watches over the web and triggers file while data leakage and sends automatic data leake information to application where the data has been leaked and further follows the previous systems. 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 leak. 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 act as a type of watermark for the entire set, without modifying any individual members.

If it turns out that an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

We start in Section II, by introducing our problem setup and the notation we use. In Sections IV and V, we present a model for calculating “guilt” probabilities in cases of data leakage. Then, in Sections VI and VIII, we present strategies for data allocation to agents. Finally, in Section VIII, we evaluate the strategies in different data leakage scenarios, and check whether they indeed help us to identify a leak.

II. Problem Setup and Notation

A distributor owns a set T \( \{t_1, t_2, \ldots, t_n\} \) of valuable data objects. The distributor wants to share some of the objects with a set of agents \( U_1; U_2; \ldots; U_m \) 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 \( U_i \) receives a subset of objects, determined either by a sample request or an explicit request:

- Sample request, Any subset of \( m_i \) records from T can be given to \( U_i \).
- Explicit request, Agent \( U_i \) receives all T objects that satisfy condi.

Suppose that after giving objects to agents, the distributor discovers that a set S 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 website, or perhaps as part of a legal
discovery process, the target turned over S to the distributor.
Since the agents $U_1, \ldots; U_n$ 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 were obtained by the
target through other means. For example, say that 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.
We say an agent $U_i$ is guilty and if it contributes one or more
objects to the target. We denote the event that agent $U_i$ is guilty
by $G_i$ and the event that agent $U_i$ is guilty for a given leaked set S
by $G_{iS}$. Our next step is to estimate $P(G_{iS})$, i.e., the probability
that agent $U_i$ is guilty given evidence S.

II. Related Work

The guilt detection approach in the previous system is related to
the data provenance problem [3]: 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 R, 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, [1,7,10,17] and other works have also studied marks
insertion to relational data.

Our proposed approach works with the previous systems and to
detect the data leakage. Finally, there are also lots of other works
on mechanisms that allow only authorized users to access sensitive
data through access control policies [2,9]. 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.

We understand the previous work nature of The Agent Guilt
model, Guilt model agent analysis, Data allocation problem,
allocation strategies. The proposed method out performs to
detect data leakages over the web compareto the other systems.

III. Proposed Technique

Sometimes a data distributor gives sensitive data to a set of
third parties. Some time later, some of the data is found in an
unauthorized place (e.g., on the web or on a user’s laptop). The
distributor must then investigate if data leaked from one or more
of the third parties, or if it was independently gathered by other
means [1].

The Detect File Watcher is a technique that works with common
protocols and the technologies to detect the data leakage over
the web. Technique that identify, monitor, and protect data in use
(e.g. endpoint actions), data in motion (e.g. network actions), and
data at rest (e.g. data storage) through deep content inspection,
contextual security analysis of transaction (attributes of originator,
data object, medium, timing, recipient/destination and so on) and

with a centralized management framework. File Watcher are
designed to detect and prevent unauthorized use and transmission
of confidential information.

The proposed technique that out performs with compare to the previous
results. The File watcher observers over databases, applications,
emails, web, FTP, servers, endpoints, print, removable media and
other custom channels. The proposed technique incorporates in
the different technologies and common protocols.

We specify the different types of files against the protocols and
technologies where the data has been leaked. The techniques
watches the entire system and detects data leakages by triggering
the file and creates an alert file for the data leakage to take further
protection the system.

IV. Experimental Results

We implemented the presented File watching technique to
simulated data leakage problems to evaluate their performance.

V. Conclusions

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 that 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 technique simple to use over
the system applications. The 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 that are not studied in this paper.
For example, what is the appropriate model for cases where agents
can collude and identify fake tuples? A preliminary discussion of
such a model is available in [14]. 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
GA has the excellence of implicit parallelism, that Holland had
pointed out that it processes mode on level of O(n^2), but in fact it
only does the calculation only on level O(n). In our experiment
system, GA do thoroughly calculation for data mining, so that
we can find out the expected and latent association rules in our
transaction database by genetic learning mode. Our system is
the data analyzing part of intelligence service of Web, and test
data includes Web log and site topology drawn out from Suzhou
University Web site built by the Institute of Intelligent information
Processing and Application.

Finally, The results of analyzing show that we can realize the
optimizing and forecasting for exist and latent association rules,
by utilizing GA to mine the statistic mining result, and it surely
can compensates halfway results that by the no sufficient visiting
amount of Web users on one certain class swarm. Thus, we draw a
conclusion that the genetic mining can catch up with this learning procedure of Web users well enough, and it is also cluster the behavior characteristics of user after processing the genetic mining.

References


