Information Behaving Badly

Julie Boxwell Ard
University of California, Davis
Davis, CA
julieard@gmail.com

Matt Bishop
University of California, Davis
Davis, CA
mabishop@ucdavis.edu

Carrie Gates
CA Labs
New York, NY
carrie.gates@ca.com

Michael Xin Sun
University of California, Davis
Davis, CA
xinsun@ucdavis.edu

ABSTRACT

Traditionally, insider threat detection has focused on observing human actors — or, more precisely, computer accounts and processes acting on behalf of those actors — to model their "normal" behavior, then determine if they have performed some anomalous action and, further, if that action is malicious. In this paper, we shift the paradigm from observing human behavior to observing information behavior by modeling how documents flow through an organization. We hypothesize that similar types of documents will exhibit similar workflows, and that a document deviating from its expected workflow indicates potential data leakage.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Unauthorized Access*; D.4.6 [**Operating Systems**]: Security and Protection—*Information flow controls*

Keywords

Data exfiltration; insider threat; workflow; data motion

1. INTRODUCTION

The goal of this work is to apply a data-driven approach to analyze information flow within an organization in order to identify workflows, the types of information that contribute to those workflows, and to characterize deviations from those workflows. In order to develop this new approach, we first characterize information behavior within an organization. The resulting insight can be leveraged to design a system capable of predicting and preventing undesirable information actions including data leakage, as well as supporting helpful information actions such as tracing the origin of effective work contributions. What we learn, by either proving or disproving our hypotheses, will prove useful for enhancing both data leak prevention (DLP) and insider threat detection methods.

DLP solutions examine the content of data, including email transmissions and activity that occurs at a network boundary point, to ensure that the data does not contain sensitive information violating organizational policy. DLP technologies typically use signatures and complex pattern matching expressions to recognize sensitive information. While generic signatures can be developed for specific domains such as the financial or health sectors, in order to be as effective as possible, the signatures need to be tuned for the specific organization [19]. By way of contrast, we examine the motion of the data to determine whether it follows its expected work flow.

Data leak detection is a specific instance of insider threat detection, in which one seeks to identify an actor (the "insider") performing an undesirable action, in this instance leaking sensitive information, whether accidentally or maliciously. In general, current insider threat detection methods model human behavior, investigating anomalies discovered in users' actions. Our approach models information behavior — not user behavior.

More explicitly stated, our approach differs from existing methods in that our focal point is the *flow of information*, not the user. We will combine the actions taken on different types of information *across* users. We seek to map the types of information used to support different steps in workflow processes and identify when the *information* itself is being used in a manner different from that which its expected workflow dictates.

This approach prompts three research questions. First, we want to know if we can derive such workflow patterns, in an automated fashion, based on how users interact with data. Second, assuming we can derive these patterns, how can we best do so? Third, we wish to characterize what deviations of data types from these patterns imply.

In this paper we present the Gemini system, which examines the movement of documents through the organization by combining file actions over time to form a document's *trajectory.* The document's trajectory represents its movement and evolution through a period of time. Gemini groups these trajectories into *patterns*, where a subset of the patterns represent organizational processes. These workflow processes represent actions that *humans* take to support their organization's tasks. Each pattern is made up of a group of trajectories, and this grouping will transcend individual users.

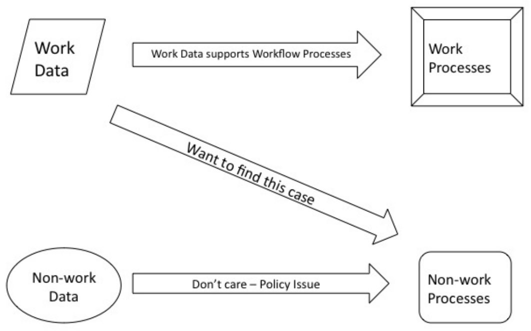

Figure 1: Overview

Gemini forms a baseline of information flow by examining the documents in an organization and applying a metric to determine their similarity based on both content and metadata. It monitors changes to existing documents and creation of new documents. Gemini then analyzes these for similarity to documents in existing trajectories and maps the documents to expected patterns. When a new document begins to move through the organization, for example via email and subsequent edits possibly by different users, Gemini flags any deviation in the document's trajectory from the expected pattern, thereby indicating potentially malicious behavior. In short, we want to identify when information mapped to work processes crosses over to a non-work process, as illustrated in figure 1.

Several assumptions underlying Gemini are outlined in Sections 4.2 and 4.4. We propose experiments to test this system and thus validate these assumptions. These experiments analyze data-at-rest in order to enable actions to protect data-in-transit. We propose using the Enron data set to model a filesystem and the workflow processes present in an organization as described in Section 4.1.

The rest of the paper is organized as follows. Section 2 describes the overall problem domain and key related work. Section 3 presents the design of a system that can extract workflows and organizational behavior by observing documents. We describe the design of an experiment to test this system, and thus each of the underlying hypotheses, in Section 4. Section 5 describes the potential impact of this system, along with its key limitations. We then conclude in Section 6 with a discussion of future work.

2. ABOUT THE PROBLEM

Of the healthcare data breaches examined in the Ponemon 2012 Patient Privacy study, 42% were caused by "employee mistakes or unintentional actions" [2, p. 2], the second most common cause.[1] Employee mistakes in handling data should be prevented by increased training, but the study concludes that HIPAA and security awareness training appears to have little positive impact [2]. Our proposed method will also apply to inadvertent data leakage, a nontrivial contributor to the problem of privacy violations [19].

[1]The most common cause, accounting for 48% of the breaches, was "a lost or stolen computing device".

Industry standard DLP products use a number of techniques to protect data in three main states: data-at-rest, data-in-use, and data-in-motion. Data-at-rest refers to data in storage, data-in-use is monitored by agents while a user accesses it, and data-in-motion is protected by network-based mechanisms [19].

Methods that detect data leakage include context-based inspection, content-based inspection, and content tagging. Context-based inspection relies on file metadata, and content-based inspection looks inside the file. Examples of content-based inspection are identifying keywords such as "confidential", fingerprinting based on string searches or hashing, natural language analysis methods, and statistical analysis including the frequency of specific terms. Content tagging indicates the nature of content through metadata, or *tags*, that are set either automatically or by a user. The content retains this tag throughout processing by other applications, and is handled according to policy defined for that tag. On the other hand, methods which prevent data leakage include access control, disabling functions such as copy-and-paste, and encryption [19].

By tying the notion of "data leakage" to the workflow that the document *should* follow, the Gemini system should have fewer false positives than current DLP methods, but may miss other types of anomalies that current DLP methods will detect. We accept this because Gemini is not an all-inclusive policy enforcement tool [21]; it is intended to augment existing DLP methods.

2.1 Related Work

Data provenance seeks to trace the lineage of data, focusing on the integrity and completeness of such records [6]. Our method uses document location records to help derive file trajectories. Our approach also incorporates file content similarity comparisons. Finally, we focus on reconstructing workflows rather than simply examining the provenance of data.

A significant problem in DLP is protecting confidential and proprietary data that exists in "unstructured form", such as source code and design data. Research in natural language processing (NLP) seeks to counter this challenge [19]. Because the Gemini system compares file similarity independent of content structure, it naturally includes proprietary unstructured data. Additionally, because DLP products focus on preventing leakage that goes outside the organization, they are deployed at network gateways and thus do not protect against intra-departmental violations [19]. Gemini is inherently able to identify and prevent such data leakage because it incorporates organizational data including users' departments and examines workflows *within* the organization in addition to data crossing the organization boundary.

Policy implementation and system training are labor-intensive initialization tasks required to enable effective DLP in an organization. These involve translating policy into rules and identifying sensitive documents on which to train the product. Periodically, organizations should revisit this initial configuration to take into account changes in policy and new types of proprietary data [19]. Gemini works independently of platforms and organizations. Gemini requires minimal user information that is easily obtained from an organization's human resources department, and will incorporate continuous learning in order to automatically update the user data in response to organizational changes. Gemini infers that a document is important by the way it moves through the organization. Existing DLP products are unable to correlate alerts to detect, for example, multiple attempts to transmit a file [19]. Because Gemini follows the data, multiple accesses and actions on similar types of data are automatically correlated by construction of the file trajectory and its participation in a pattern.

Insider threat analysis focuses on modeling individual behavior. For example, a company named Cataphora has patented a method of analyzing "Discussions" by tracking communication across several media, forming a document (communications) family tree, deriving actor roles, and from that analyzing the workflow on an *ad hoc* basis. Their analysis focuses on the actors: determining influential employees, deriving workflows based on the roles of the actors, and tagging the workflows based on the roles and departments of the *actors* [1]. Our analysis focuses on the *data*: we form document trajectories based on the actions of individuals and then generalize this activity into patterns to determine whether a *document* deviates from its expected pattern.

The Enron Email Corpus is a valuable dataset for academic research, primarily because it consists of real-world employee exchanges from a large organization. Researchers have used it to study email classification [13], social networking [15], and as a business case study [16]. A method of inferring the corporate hierarchy is presented in [4]. Analyses of the files attached to Enron email are sparse; Dredze *et al.* [8, 7] use email text to predict whether an email should have an attachment.

Personal Information Management (PIM) focuses on how workers can organize information so that the information they need is readily available [12]. Anomalies in audit logs and information flow have been considered as indicators for intrusion detection [22, 5].

Role mining seeks to define a role-based access control (RBAC) matrix based on user permissions [9, 14, 18]. Our approach differs from role mining in two significant ways: we seek to identify workflows—a sequence of actions—whereas role mining groups sets of permissions assigned to users. Additionally, the focal point of our approach is the *information*, not the user.

The body of work on process mining provides algorithms to derive business processes from user actions captured in event logs [25]. Conformance checking is an analytic method seeking to identify differences between the process definition and process execution in reality [23]. The benefits of conformance checking to internal auditing are many, including fraud detection, monitoring required separation of duties, and process improvement [11, 3]. The informational perspective of process mining highlights the relationship between the data consumed and produced during execution of business process tasks, but there is scant literature on exploiting this perspective of business process mining [17]. In the language of business process mining, our approach seeks to apply conformance checking using the informational perspective. We will leverage business process analytical methods as appropriate in our analyses.

3. SYSTEM DESIGN

Gemini is essentially an anomaly detection system applied to workflow. Gemini will *infer* workflows based on file activity because defined workflows often do not match execution in reality [24]. Like other anomaly-based systems [21], it requires a training phase to establish the "expected workflow" of documents. The input data is information about the evolution of the files over time. Additionally, not all anomalies will be considered bad data behavior [10]; we will use the results of this experiment to more tightly constrain the types of anomalies that merit investigation. The training phase takes three steps:

1. For each file, record the metadata and actions associated with operations performed on the file. These actions are placed into a temporal sequence called a *trajectory*.

2. Group the trajectories into sets of "similar" trajectories. "Similar" is defined by a *similarity metric* that takes both content and metadata of the files for each trajectory into account, because looking at both often points to relationships among trajectories that may not be apparent from either examined separately.

3. Use the evolution of the files, and the similarity metric, to determine when files have been moved or edited, and add these changed files to the files' trajectories.

The term "evolution of the files" refers to data that shows the changes to the files and to the file system over time, such as snapshots of the file system arranged in temporal orde. Other data sources may provide the needed information, for example a software agent that records changes to data as it is used and as it is in transit.

A *pattern* is an abstraction of a group of trajectories that captures organizational relationships and periodic actions. We expect that different files following similar trajectories will exhibit other similarities as well, such as in the content of the file, or the departments of those users who created the file. By noting the similarities among the files at the start of similar trajectories within the same pattern, we can determine the common characteristics of files in that pattern. We expect that a subset of all possible patterns will map to workflows (organizational processes). The remaining patterns that do not describe organizational processes will not indicate anomalies to the Gemini system, and so we can ignore them.

During Gemini's deployment phase, when a file is first created, its content and metadata are compared to those of files at the base of every already-identified trajectory. Similarly, when an already-analyzed file undergoes additional actions after the baseline is established, the modifications to its trajectory are compared to those of similar files in other existing trajectories. A file might be similar to files in multiple patterns. As a new file moves through the system, its trajectory is compared to the trajectories of similar files and to established patterns until the file's trajectory is mapped to one or more existing patterns. As the file's trajectory matures, the expected pattern mapping will narrow down to a single expected pattern. If the file then deviates from this expected pattern, Gemini will generate a response appropriate to that deviation. Because the cost of both false positives and false negatives are high, we propose a staged response taxonomy, elaborated in section 3.4, to guide the system's response. The narrow scope of our analysis, as well as the increased granularity of the information analyzed, will enable the system to decide on the response appropriate to the situation.

In a production scenario, Gemini will incorporate a continuous learning function to update the patterns over time to take into account users changing departments, new users, user and departmental reorganizations, new workflows, changes to existing workflows, and other organizational changes.

3.1 File Similarity

There are a number of ways to calculate file similarity. Plagiarism detection software estimates the extent to which one document is derived from a pool of other documents. Automatic keyword identification is a more content-specific option that inspects the intersection of groups of keywords to calculate similarity. Natural language processing (NLP) methods account for synonyms and seek to extract meaning from text. This portion of Gemini must be a module, so that the implementers can select any similarity metric that they think effective in the environment in which Gemini is used.

Initially, we explored using a proprietary algorithm developed by CA Technologies called *Affinity*, which indicates the extent to which content is shared between two documents. This enables bidirectional comparison. CA Technologies' algorithm may be tuned to match differing levels of fidelity from the page level down to the paragraph or sentence level. Affinity successfully detects relationships between files of unstructured data such as ordinary text or source code.

The Affinity algorithm outputs two percentages that indicate the extent to which document A was derived from document B, and *vice versa*. This bidirectional similarity is needed to detect the case where a one-page A is completely contained in a 100-page B. In this case, A is 100% similar to B, but B would only be 1% similar to A. One can combine the two percentages in some way, for example averaging them or taking the maximum, to obtain a single number if desired.

We plan to investigate different thresholds and combinations of metrics, and how the environment impacts their effectiveness. For example, taking the maximum of the two percentages might indicate whether a one page document is a subset of a one hundred page document. Such a relationship may be lost if the two numbers are simply averaged. We will also examine the performance and privacy trade-offs involved with analyzing file content. For example, how does the combination of file content and file metadata, versus simply using metadata, benefit the identification of workflows and deviant files? We expect that analyzing content similarity will reveal relationships between files with different names. However, we do not know often users rename files. Additionally, analyzing content similarity will likely identify relationships *between* types of information, such as when data from different sources are combined into one document.

We will also consider methods that improve the performance of a deployed system. For example, when adding files to the corpus, it may not be necessary to compare every file to every other file. Perhaps new files need only to be compared to selected files which represent groups of similar files.

3.2 File Trajectory

A file's trajectory consists of a sequence, ordered by time, of actions taken on it by users. The trajectory begins with the first action observed, and ends either when the data stops moving, is deleted, or leaves the organization.

We consider file manipulation actions that are independent of the underlying operating system. This allows us to create a general model. We examine *actions* that a user could take on a file rather than the traditional system permissions of read, write, and execute. Given the environment we are testing our system in (see Section 4), we cannot model execution of a file, but such a model would be useful to track a binary file's trajectory, enabling for example the identification of malware spreading within a network. We therefore focus on actions that we can discern from file reads and writes, and so define the following file actions for our model: *create*, *read*, *write*, *delete*, *copy*, and *edit*. A user and a timestamp are associated with each of these actions for a given file.

We call *create*, *read*, *write*, and *delete* "first-order" file actions because they are determined on a single pass through the data. The first time (chronologically) a file appears marks its "creation". We determine this time by analyzing file metadata and file similarity. When an exact copy of a file appears elsewhere with no modifications, we define that as a file "read". That is, if Tom has file X at time t_1, and Kathy has file Y, which is an exact copy of X, at time t_2 where $t_1 < t_2$, then Kathy has read X. A file "write" occurs any time a modification is made and the resulting file is similar to the previous version (within some similarity threshold). For example, if Tom appends data to, or deletes data from, file X at time t_3, then he has written X. Finally, a file is "deleted" when the file exists at time t_1 but not at time t_2, where $t_1 < t_2$.

Copy and *edit* are "second-order" file actions because they are identified on a second pass through the data, after identifying the first-order file actions. Two successive reads by different users constitute a "copy" action. Suppose Tom reads file X at time t_1, and Kathy reads X at time t_2 where $t_1 < t_2$, and the metadata of X reveals a link between Tom and Kathy (say, they are members of the group that owns X). Then Kathy is said to have copied X from Tom. *Edit* is defined as a read followed by a write. Note that the users involved need not be distinct; that is, Tom could read file X twice in succession to copy it.

In formal terms, associated with each file action are a user, an action, and a timestamp of the action. Here, "users" includes shared resources such as a database. A user may belong to one or more organizations, of course.

For each user u, we have the following characteristics:

$$u = \{username, organization, job\_title\}$$

For each file f, we have the following characteristics, in addition to other dataset-specific metadata:

$$f = \{name, extension, size, ftype, location\}$$

Let f represent a file and u represent a user. An action a on a file f by user u at time t is represented as $a(f, u, t)$. We combine actions into a linear sequence

$$a_1(f_1, u_1, t_1), \ldots, a_n(f_n, u_n, t_n)$$

over the life-cycle of the file f, based on the timestamps, to form the file's trajectory.

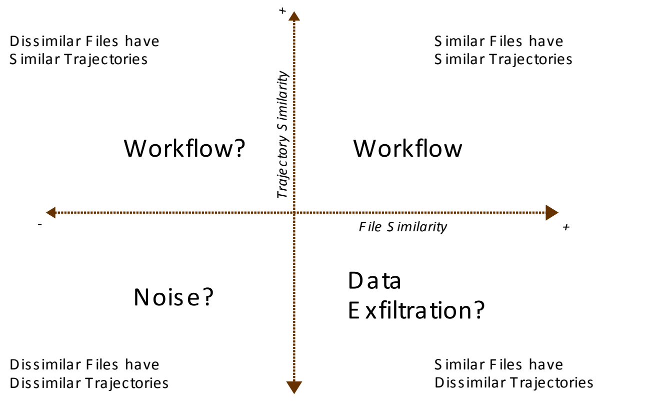

Figure 2: The File vs. Trajectory Similarity Matrix

Note that the trajectories of two similar but different files may involve different times. For example, suppose two people must file a report by time t. The first user's document may have timestamp $t_{first} \ll t$, and the second user's document may have timestamp $t_{second} = t$. The critical observation for similarity of trajectories is that both are completed by a specific time, t. For notational convenience, we will represent the two actions as having the same timestamp t, because it makes exposition simpler. But the reader should bear in mind that two files may have similar trajectories even when the timestamps associated with some actions differ across the trajectories. The similarity metric must deal with this difference.

Trajectories are grouped based on their similarity and organized into patterns. Ways to explore trajectory similarity metrics include graph similarity measures and machine learning techniques.

3.3 Patterns

A group of similar trajectories forms a *pattern*. Because trajectories consist of actions that humans take on a file, the patterns will also represent workflows performed by humans. In addition to the ordering of actions given by the timestamps, we consider the duration of the interval between actions, and the actions' frequency. These become a characteristic of the trajectory, and we can identify periodic processes such as weekly, monthly, or quarterly reports.

Most organizations have many well-defined processes such as filling out timesheets and filing and paying invoices. The steps include preparation of the documents and a well-defined approval chain. An individual employee will submit his timesheet to a supervisor. The supervisor may approve it and forward it to payroll, or reject it and require the employee to submit a correction before forwarding it to payroll. Information that goes into an invoice may be derived from purchasing and timekeeping records, then routed to management, and finally to contracting.

Less well-defined processes may include writing code and technical reports. In these processes, members of a team exchange multiple copies of drafts, receive feedback that results in modifications, and then create a final submission. Other workflows may depend on an approval the content of which is not sufficiently similar to other information in the workflow, and in such a case, trajectory similarity may be more important than file content similarity. We acknowledge that some types of legitimate workflows may not follow regular patterns, and one contribution of this experiment will be identifying the types of workflows for which this method is effective.

The actions in the trajectories will undoubtedly have different timestamps, but as noted above the timestamps will be close. Aggregating the information will indicate similarities in the trajectories, such as all employees sending their timesheet to management before 5pm on Friday, management approving or rejecting them before 5pm on Monday, and payroll emailing paystubs to each employee by 5pm on Wednesday. These similarities form the basis for a pattern.

An interesting question is how to handle files with metadata and content that are not similar to those of any other files. Every file has a trajectory; the trivial case is when there is exactly one occurrence of a file, its trajectory consisting of a single action ("create"). For such files, we focus on the user who created the file. The other trajectories associated with this creator may enable us to form some conclusion about why these files are not part of some other pattern. We are also interested in any trajectories that differ from all other trajectories. Understanding these special cases will help us categorize the "noise" quadrant in Figure 2.

Perhaps graph similarity metrics or social media flow patterns will suggest a method of automatically grouping trajectories into patterns; we plan to experiment with various thresholds to test this idea. Patterns representing business functions will be categorized as workflow processes. This will be organization-specific and we will seek methods to automate this.

3.4 Responses

Current DLP responses consist of blocking a transmission or sending an alert. In order to effectively block the transmission, one must have a high level of confidence that data leakage is occurring; otherwise, a false positive effectively produces a self-inflicted denial of service attack. The latter response, sending alerts, can produce false positives that requires a human analyst to detect.

A more detailed taxonomy of responses can describe more fine-grained automated responses and the situations in which they are appropriate. The goal of the response taxonomy is to use the insight gained from analysis to reduce the number of false negatives as well as to prevent self-inflicted denials of service resulting from false positives. For example, if some anomalous activity could be data leakage, then the system could increase monitoring and analysis of all accesses to the appropriate file or files. If data leakage occurs through an email attachment, the system could replace the restricted file in the attachment with a bogus file or strip the attachment, and then send the email on. If data leakage occurs in the body of an email, the system could encrypt or redact only the restricted information, or replace it with nonsense text. If large amounts of essential data are being deleted from the file system, then the system could automatically copy the information to a backup repository instead of simply disallowing the "delete" action. These options are depicted in Figure 3.

The system could also examine the context of the suspect action relative to other actions to determine whether it should take additional response measures after blocking a user action. The user's response to the response could also

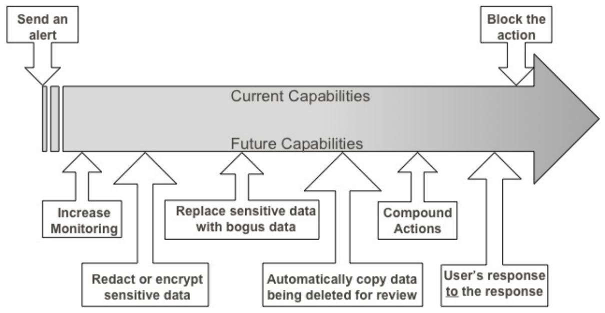
Send an alert
Block the action
Current Capabilities
Future Capabilities
Increase Monitoring
Redact or encrypt sensitive data
Replace sensitive data with bogus data
Automatically copy data being deleted for review
Compound Actions
User's response to the response

Figure 3: DLP Responses

feed back into the system. For example, if someone makes a second attempt to transmit the information, one possible reason is that the user may not realize the violation of policy. In this case, the managers need to clarify the policy or ensure all employees know it. The deployed system could also incorporate warnings, asking a user if they really mean to perform an action, in order to prevent leakage before it occurs.

The reader will recall that our adversary includes intentional, inadvertent, internal, or externally-initiated data leakage threats. Adversarial capabilities as well as common employee mistakes will inform the types of workflow deviations to which Gemini should respond.

4. EXPERIMENTAL DESIGN

This section presents the hypotheses to be tested, and the experiments to be run. We begin with a description of the data set, and then discuss the abstraction of trajectories and patterns from the file system. We can then state our hypotheses, and present the experiment.

4.1 Dataset

In order to enable reproducibility, we use the Enron Email Corpus dataset. This dataset is from a major U.S. corporation that collapsed, and in the ensuing fraud investigation the U. S. Securities and Exchange Commission (SEC) seized a large corpus of emails for use at trial. These emails were subsequently made available by CMU and then Amazon Web Services. Note that Amazon Web Services has since removed the Enron Email Corpus due to concerns about user privacy.

The CMU-hosted dataset is frequently used in email and social network research. However, the CMU-hosted dataset has been preprocessed to resolve duplicate aliases and contains 517,431 emails with no attachments. Several other versions of the Enron Email Corpus have been made available by different researchers and have been processed to support their research focus. They consist of varying numbers of emails, but most of them do not contain attachments, and of those that do the highest number of attachments we found was approximately 41,000 attachments. The Amazon Web Services version contains 1,227,255 emails with 493,384 attachments and totals 210 GB. Since our initial analysis focuses on the attachments, we processed the original dataset in order to obtain all email attachments and information about the emails to which they were attached. To our knowledge, this is the first exploration of all versions of all attachments in the Enron Email Corpus.

Amazon Web Services hosted the Enron Email Corpus in several formats. We chose to analyze the XML version. Our initial analysis showed that a large number of attachments are URLs; that is, if a hyperlink is pasted in the email body, information about the link is included in an attachment. We have identified several common attachments such as "mime001.txt" files, vcf cards, and "winmail.dat". As our similarity metric uses text, all binary files including executables are excluded from our analysis. We set aside all attachments other than text ones, including all media files including audio, video, and image files, for later analysis.

Nearly half of the files are Microsoft Word documents. An additional 15% are Excel spreadsheets, and 7% are in Portable Document Format (PDF). If the PDF was created by conversion from a word processor document, our file similarity metric will be able to analyze its content. Several other word processor formats are present, including ".wpd" (Word Perfect file), as well as additional text formats such as ".txt", and ".rtf" (Rich Text Format). A nontrivial 6.5% of the files have no extension. Initial inspection of these files shows that several of them do indeed represent relevant documents. We will explore an automated method for adding the appropriate file extension to this set of nearly 31,000 files so that our similarity metric can process them. There are also over 1,400 zip files, which we plan to extract and add to the constructed filesystem.

A *custodian* is the user who owns the email files under consideration. There are 151 total custodians, one quarter of whom are identified only as "employees". Eight of the users (5%) hold the title of either CEO or President, while twenty-eight (19%) are Vice Presidents. There are twenty-three directors (15%), thirteen managers (9%), and five managing directors (3%). A full twenty-nine (18%) do not have a job title [20].

For our initial analysis, we grouped files with exactly the same name as similar and sought the relationships between them based on their metadata. This initial grouping method will be replaced with file content similarity groupings when we finish implementing a file content similarity metric to identify groups of similar files.

The initial analysis revealed that duplicate files are very common. The reason for this is that the same file appears in multiple folders. For example, files with exactly the same metadata will appear in a particular custodian's Inbox, Discussion Threads, and All Documents folders. We are currently incorporating duplicate removal in to our algorithm. If a file has the same custodian and timestamp, we arbitrarily choose one version to retain and remove the others from consideration. The impact of duplication is nontrivial; in one analysis, we found only 69,193 distinct files in a group of 230,116 metadata records after removing 160,953 duplicates. In this instance, only 30% of the files analyzed were distinct according to our duplication removal algorithm.

Another initial finding is that when the sender carbon copies themselves, this can confuse trajectory construction. We will try removing a recipient from the recipient list when that recipient is also the sender. Another complicating factor is inconsistent aliases. The user Matt Bishop may be represented by "M Bishop", "Matt Bishop", "Matt A Bishop", "Bishop, Matt", "Bishop, Matt A", "bishop@ucdavis.edu", "bishop@cs.ucdavis.edu", and other variations. We are currently working to resolve aliases and point variations on one custodian's name and email address to the same user.

We suspected that patterns will link various departments along a workflow process, so we searched the literature for an organization chart that would describe the relationships custodians had with each other. Unfortunately, we were unable to locate such a chart. Recall that [4] attempts to infer Enron's corporate hierarchy by analyzing email content. Fortunately, our initial analysis of the dataset found several promising leads, and so we are attempting to derive a chart from the initial count of 1,216 attachments containing the words "org" and "chart" .

4.2 Modeling a File System

In order to model a functioning business, we needed to create a file system using the email attachments. Thus, we assumed that documents attached to email represent a subset of a user's files. So we use the email metadata to develop our file metadata. For example, suppose document doc_1 is attached to email $email_1$. Associated with $email_1$ is a sender $sender_1$, recipient $recip_1$, and timestamp t_1. Suppose that $action_1$ is associated with $email_1$. Then the file doc_1 has the following properties:

Filename: doc_1

Time of file action: t_1

Users associated with some file action: $sender_1$, $recip_1$

Emails and their attachments map into first-order file actions as follows. The first time (chronologically) an attachment doc_1 appears, we say that $sender_1$ created doc_1. When $recip_1$ receives doc_1 as an attachment, we say that $recip_1$ read doc_1. Carbon-copy (CC) recipients as well as primary (TO) recipients all read doc_1. When $sender_1$ transmits a file similar but not identical to a previously seen version of doc_1, we say that $sender_1$ writes doc_1. When doc_1 is attached to an email found in $recip_1$'s deleted items folder, then we will say $recip_1$ deleted doc_1.

Data Field	**Description**
DocID	A unique value for each file that also references the email to which it was attached.
From	The email sender
TO	The email recipient(s)
CC	The carbon copy email recipient(s)
Subject	The subject of the email
Date sent	yy-mm-dd
Time sent	hh24-mm-ss
AttachmentCount	The total number of file attached to an email
AttachmentNames	A complete list of attachments
Custodian	The Enron employee whose emails we are analyzing
LocationURI	The email folder in which the email is stored
FileName	The attachment filename
FileExtension	The attachment filename extension
FileSize	The size of the attached file

Table 1: Metadata retained for each file

Emails and their attachments map into second-order file actions as follows. Suppose that $sender_1$ emailed doc_1 to $recip_1$ at time t_1. As above, we say that $recip_1$ read doc_1. If, some time after t_1, $recip_1$ emailed a file identical to doc_1 to $recip_2$, then we say that $recip_2$ read doc_1 and that $recip_1$ copied doc_1. Suppose that $recip_1$ read doc_1 at time t_1, and then transmitted a file similar but not identical to doc_1. We say that $recip_1$ edited doc_1.

We assume that the email timestamps are from the same server. The filesystem is structured in the following manner to associate each attachment with a unique file path:

Custodian/email_subject/email_folder/yy-mm-dd/
hh-mm-ss/FileName.ext

This structure consolidates activity by file name for each custodian while preserving uniqueness down to the second. We also formed a mirror directory where metadata for each file is preserved in a "metafile". This simplifies accessing metadata during file analysis. We should note that the deployed system will not depend on the filesystem structure. For example, we considered organizing the attachments as follows:

Custodian/email_folder/FileName_timestamp.ext

We decided to use the first structure in order to preserve the integrity of the original file names. Because similarity is determined among all files, the construction of trajectories and patterns is independent of the filesystem structure. This approach will enable Gemini to be deployed in any organization, on any user's filesystem, regardless of how the user organizes their files.

We preserve the metadata fields listed in Table 1 in the metafile.

4.3 Generating Trajectories

Generating the trajectories for the Enron dataset poses a challenge. As described in Section 3, when Gemini starts, it develops trajectories as the system evolves. But the Enron data is static, so there is no evolution of the system. We need to simulate such an evolution. To do this, we modify the three training steps as follows:

1. Group the files into sets of "similar" files. "Similar" is defined by a *similarity metric* that takes both content and metadata into account, because looking at both often points to relationships among files that may not be apparent from either examined separately.

2. For each set of similar files, derive the actions that relate the different files to one another. These actions are placed into a temporal sequence called a *trajectory.*

3. Use the evolution of the files, and the similarity metric, to determine when files have been moved or edited, and add these changes to the files' trajectories.

Here, we examine the existing files and their associated metadata to derive the changes made to the files. We assume that similar files are modified variants of one another. This assumption is necessary to develop the trajectories.

To generate the trajectories, we use a file content similarity metric, comparing every file to every other file. This produces a list of the files and two similarity matrices reflecting the bidirectional similarity of the files. The organization of our filesystem (described above) ensured the file paths were unique, unless the same filename appeared in the same email folder of the same custodian more than once per second.

We want to identify files that may be related and determine whether they form one or more trajectories. Files that are very similar will most likely be related. We will experiment with various similarity thresholds, beginning with 100% (identical) and decreasing the threshold, to group similar files. Recall that we compute separate bidirectional similarity metrics. The minimum, maximum, and mean of these two numbers for each file will provide measures of similarity. Grouping files by content similarity should identify identical or similar files that have different filenames.

In addition to the similarity metrics, metadata correlations may help us group files belonging to the same trajectory. The metadata contains both file metadata and information about the email to which it was attached. The file metadata includes the file name and size, while the email metadata includes the sender, recipient(s), carbon-copied recipients, the subject, timestamp, and number of attachments. As one would expect, records of blind carbon copies (BCC's) were not preserved in our dataset.

Once we find a group of files that are likely related (under the similarity metric and metadata correlation), we will identify the first-order and second-order file actions to form the data points described above. For example, there are 160 emails with files named "Resume.doc" attached. By applying the similarity metrics to the content and metadata for these files, we may find several completely different resumes. We want to identify edited and renamed copies of resumes. Even if "Resume.doc" were renamed to "Julie.doc", the similarity metric would show the connection between these documents. We then identify the file actions, form the data points, and combine them into trajectories based on the timestamps of the file actions. Looking at the users and email subject associated with a particular file, we can tell if there are separate, forked trajectories, or how the trajectory may have evolved. We will analyze a forked trajectory as a graph, possibly separating the forks and considering each part from start to finish. We may also find that some of the files that are grouped together due to content turn out to be unrelated because the file actions are different, or tine stamps do not match up sufficiently.

The file metadata will also preserve the fact that two files were attached to the same email when that is the case. This may indicate that these "co-attachments" follow similar trajectories, or are part of the same pattern. Even if a file appears once and is not similar to any other files, it will have a minimum trajectory of being created by one user (the sender) and read by the recipient(s).

We began by seeking "chain-of-custody" trajectories, where one can identify the flow of information from $user_1$ to $user_2$. However, in the absense of complete data, we will instead log accesses to the data that appear in our dataset. While the Enron Email Corpus consists of presumably complete records for each custodian during the specified timeframe, there may be other Enron employees whose records would complete these "chain-of-custody" trajectories. Without those employees records, this algorithm did not find many lengthy trajectories. Additionally, the "chain-of-custody" approach requires the resolution of different aliases pointing to the same user, otherwise this algorithm would not identify the flow. An example of this is when $user_1$ sends information to $user_2$, but $user_2$'s mail folder saved $user_2$'s alias in a format different from that saved in $user_1$'s mail folder.

Figure 4 depicts the frequency of different "chain-of-custody" trajectory lengths within the group of 230,116 metadata records not sorted by filename and without aliases re-

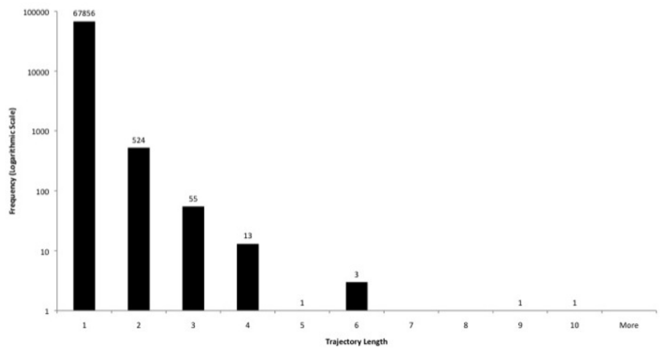

Figure 4: Trajectory Length Histogram

solved. Not pictured on the logarithmic y-axis are the single instances of trajectory lengths 5, 9, and 10. As the reader can see, the vast majority of files have a trivial trajectory of length one, but there are several trajectories we discovered with a length greater than one, detailed in Table 2. [2]

Table 2: Trajectory Lengths and the Number of Occurrences

Trajectory Length	Count
1	67856
2	524
3	55
4	13
5	1
6	3
7	0
8	0
9	1
10	1

4.4 Abstracting Patterns from Trajectories

After we construct the file trajectories, we will explore metrics for comparing trajectories and grouping similar trajectories. This will allow us to identify attributes of patterns, and develop metrics for determining whether a particular trajectory belongs to a particular pattern.

Consider an example of a simple workflow, such as submission of timesheets on a weekly basis. Suppose that several employees edit their timesheets several times per week. Then they send their completed timesheets to their manager during a particular time interval. The manager can either approve individual timesheets or return them for corrections. In the first case, the manager sends the approved timesheet to payroll. In the second case, the manager sends the timesheet back to the employee, who makes another edit and then resubmits it to management, who will then approve it and forward it to payroll. Payroll employees process the timesheets and send emails containing paystub information to the employees.

[2]We are exploring the cause of the discrepancy between 69,193 distinct files and 68,454 trajectories generated. Each file should have a trajectory, even if it is trivial. This may be due to incomplete file metadata.

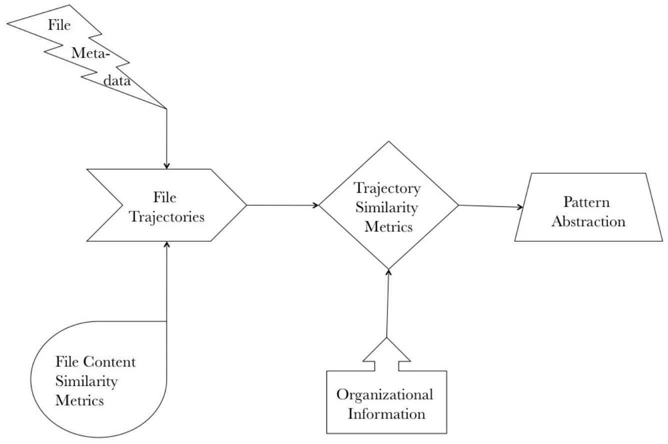

Figure 5: Analysis Flowchart

We apply the concepts of file similarity, trajectory similarity, and pattern abstraction to this example to demonstrate how our model works. From the file similarity metric, we discover similarities between different employees' timesheets, the copies received by the manager, and the copies received by payroll. File metadata will give us separate paths for each employee's timesheet, and from that we construct the separate trajectories for each employee's timesheet. Ultimately, these timesheets follow the same basic trajectories (employee, manager, payroll, email), with some variations. This creates a pattern. An overview of this process is illustrated in Figure 5.

After manually working through a few specific cases, we will automate the process of forming trajectories, refining as needed based on the results. We will then determine which patterns support workflow processes by inspection, and explore methods for automating workflow identification. We expect to gain insight into the types of information flow by examining patterns. Once we establish which patterns represent workflow processes, we will investigate any deviant files, similar to files that follow workflow pattern, but do not themselves follow that workflow pattern. We will also characterize the types of files present in our dataset which do not follow a workflow process.

4.5 Assumptions and Hypotheses

Our assumptions are as follows:

Assumption #1: The majority of activity in the dataset is work-related. The Enron Email Corpus does contain some personal correspondence, but personal correspondence typically does not have multiple versions; such attachments and emails would tend to be single instances. Because the nature of the data requires that we focus our analysis on file attachments that have multiple versions and multiple instances, most of this activity will be related to workflow processes. We will verify this by examining the content of the files.

Assumption #2: All timestamps are from the same server, so that the timestamps from different time zones, if any, have been resolved by the exchange server. We will order the file actions by timestamp, and look for conflicts to verify or refute this assumption. That is, if the file similarity metrics dictate a trajectory different from that indicated by the timestamps, we will modify the trajectory and perhaps adjust timestamps for groups of users located in a different time zone.

Our hypotheses are as follows:

Hypothesis #1: There are consistent patterns of information flow within an organization. If we are unable to discover linkages between documents exchanged by the Enron employees or if we cannot group trajectories in to patterns because the evolution, as it were, of any one file has nothing in common with the evolution of any other file, then we will conclude that consistent patterns of information flow are not represented in the Enron Email Corpus.

Hypothesis #2: Similar files will exhibit similar patterns of flow. Suppose that $user_1$ creates doc_A and $user_2$ creates doc_B, and doc_A is similar to doc_B. Then we expect doc_A and doc_B to follow the same pattern. If they do not, then this second hypothesis does not hold.

Hypothesis #3: Any file similar to other files in a workflow pattern that deviates from that workflow pattern indicates possible data leakage. If doc_A, doc_B, and doc_C are similar, and doc_A and doc_B follow the same workflow pattern but doc_C does something different, then we need to follow doc_C and determine whether the workflow is that of information being misused. This hypothesis depends on the assumption that the majority of information being exchanged at work is business related (Assumption #1).

4.6 Validation of Approach

We may encounter several types of false positives and false negatives in the course of this investigation. If two files are said to be similar when they are not, or do not meet the similarity threshold but we have some other link between them, we will revisit our similarity methods or define an acceptable error rate. Recall that file similarity will be based on both content and metadata analysis. The file content comparison will compare sentences at the most fine grained level (and not words or phrases). If for instance one person copied another person's work but changed minor words in every sentence, then the content similarity metric may not reflect this relationship.

A false positive for trajectory similarity will occur if two trajectories are said to be similar, but on inspection we determine that they are not. An example of this is if comparison of two trajectories indicated similarity, such as a timesheet trajectory and a resume trajectory. However, they are for different employees in completely different departments and the duration between file actions are completely dissimilar. On the other hand, a pair of trajectories could fail to be matched when they are actually similar. We will closely examine whether all employees timesheets follow similar patterns, for example.

We may see two files and their trajectories grouped into a pattern, but not notice a link between them. If a timesheet and a resume are grouped into pattern pat_1, while a timesheet and an invoice are grouped into pattern pat_2, we may have to include department identification into our pattern definition. If, however, two files and their trajectories are not grouped into a pattern but there is a clear linkage between them, we will need to revisit our analysis or define an acceptable error rate. One example of this would be if the timesheets of two employees in the same department end up in different patterns.

5. IMPACT, LIMITATIONS & FUTURE WORK

Our information flow analysis could be used to trace the origin of a piece of information, such as who originated a technical report or an incorrect bit of information. Learning which types of information originate from outside the organization or have an outside destination, and which patterns or workflow processes use such information, will help us develop our model of workflows. They will also tell us what types of information are required to complete workflow processes, and which workflow processes rely on shared information.

The results of our analysis will provide insight into organizational workflow processes and how departments interact with each other. Just as it can detect data leakage external to the organization, it should also detect interdepartmental data leakage or organizational conflicts of interest. This method could also identify the organizational assets required to support workflow processes, such as information that could support workload balancing or network defense priorities. The organization could use the results to improve employee training as well. Mapping information to business tasks and workflows supports the goals of Personal Information Management (PIM), which seeks to understand how people use information and improve its availability in support of such tasks [12].

We also want to identify bidirectional or one-way information flows between departments. Knowing such flows will help us identify the types of activity associated with files that deviate from expected patterns and whether data leakage can be identified this way. We also plan to characterize the types of information common to a particular pattern, particularly when dissimilar documents follow the same pattern. This type of flow could indicate multiple data types supporting the same workflow.

This method can be extended to incorporate additional sources of organizational data to enrich the definition of workflow processes. It can also be applied to additional types of data-in-transit such as instant messages and unencrypted network traffic, as well as data-in-use (such as copy-and-paste) with the addition of a software agent. The method of deployment will depend on how the organization stores information. For example, if an organization uses a cloud-based architecture to store data, then we could add cloud provenance data to the trajectory analysis. For database storage, we could leverage access logs to enhance the trajectory algorithm.

Additionally, automating data leakage prevention responses contributes to user privacy because the system assigns an action based on established criteria, instead of the intervention of a human analyst. Automatic responses will also prevent more incidents of data leakage before they occur, and having more response options available will prevent self-inflicted denial of service because options other than blocking an action will be available.

If successful, this data-oriented approach will not require user behavioral information. Instead, it follows the data and identifies data that is not following an expected workflow pattern. This data-driven approach could augment or replace current user-driven approaches. This shifts the paradigm of current insider threat investigation methods. Instead of actions flagging a user's entire account for investigation based on particular work habits that differ from other employees' work habits, specific user actions that deviate from our narrowly defined problem would generate automatic responses. An aggregated list of such triggers and responses would enable the human analyst to reduce the set of people whose behavior must be investigated to establish whether this is an attempted data leakage. Additionally, if the same deviations occur from several user accounts, this could indicate the same actor using different user accounts.

However, this approach has limitations. As this approach is highly data-driven, we can only verify this method against threats that can be modeled by our data. Data exfiltration methods that rely on communication methods such as instant messaging, personal email, and copying and pasting restricted information in to an open document and then printing cannot be represented in this dataset. Nevertheless, verification of our hypotheses will lend insights towards how information flow could be applied to those additional avenues of data flow. In future work we plan to analyze binary data in order to study how malware spreads within a network and develop methods of identifying and preventing such spreads. As this method analyzes a filesystem model which includes deleted files, this method may not be effective against highly technically savvy adversaries who know how to cover their tracks. Additionally, this method works only on plaintext and would not be effective in analyzing encrypted files or network communications.

Our future work will incorporate the Enron email bodies. We took the step of preserving this information in the same format as the email attachments. We could treat the email memos as files as well, or adapt our analysis to email. This approach could detect changes to replies and forwarded emails.

Upon completion of the Enron Email Corpus experiment, we will automate several aspects of our analysis and propose a deployment within CA Technologies. The deployed system will run incrementally on a daily basis after the initial baseline training period.

In our future work, we will consider additional file similarity calculations, such as automatic keyword tagging. One could consider the percentage of matching tags between two documents once tags were properly grouped, and a distance measure had been defined. We will also consider applying more complex social networking information as an overlay on the user information, so that the social network of a user might act as an additional characteristic to supplement organization and job title. We will also explore applying human behavioral analysis to characterize the data behavior patterns that we identify.

As mentioned above, a software agent could monitor data-in-use actions such as copy-and-paste, local egress points such as printers, device drivers for USB/CD/DVD, local host logs, database audit logs, and cloud provenance. We would like to explore methods to analyze binary files in order to detect the activity and trajectories associated with automated malware or polymorphic worm spreading. This method could be applied to rich media as well, using image, audio, or video similarity metrics instead of text similarity metrics.

In future experiments, we will look for an example within the data to model a data leakage event. We may find such an event in the Enron Email Corpus after further examination.

6. CONCLUSION

We have designed a system for detecting data leakage. That system is based on the automated modeling of document flows and organizational processes. This approach represents a paradigm shift from traditional data leakage prevention approaches, which are based on keyword analysis of data-in-motion, and traditional insider threat detection approaches, which are based on modeling human behavior. The new paradigm is based on detecting information deviating from its expected behavior, and identifying which deviations indicate bad behavior. Our response taxonomy also will reduce the chances of an automated response causing denial of service.

In addition to proposing a new paradigm, we are also testing it against a new data source. This is the first analysis, to our knowledge, of all copies of the Enron email attachments.

Finally, this "follow-the-data" approach shifts the paradigm to assist security analysts to identify the users to investigate, rather than simply investigating all anomalous activity as a potential threat.

Acknowledgments: This work was supported by Awards CCF-0905503, CNS-1219992, CNS-1219993, and OCI-1246061 from the National Science Foundation to the University of California at Davis. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation or of the University of California at Davis.

We thank Prof. Sean Peisert of the University of California at Davis and Lawrence Berkeley National Laboratory for helpful comments and discussions, and Dr. Tom Longstaff, Technical Director of the Department of Defense Office of System Behavior Research, for informal discussions that contributed to the system outlined in this paper. Peter Spellward of CA Technologies designed the Affinity document comparison metric.

7. REFERENCES

[1] Cataphora technology: Tackling the hard problems.

[2] Third annual benchmark study on patient privacy & data security. Technical report, Ponemon Institute LLC, December 2012.

[3] Rafael Accorsi and Thomas Stocker. On the exploitation of process mining for security audits: the conformance checking case. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 1709–1716. ACM, 2012.

[4] Apoorv Agarwal, Adinoyi Omuya, Aaron Harnly, and Owen Rambow. A comprehensive gold standard for the enron organizational hierarchy. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 161–165, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[5] Mohammed I Al-Saleh and Jedidiah R Crandall. On information flow for intrusion detection: What if accurate full-system dynamic information flow tracking was possible? In *Proceedings of the 2010 workshop on New security paradigms*, pages 17–32. ACM, 2010.

[6] Peter Buneman, Sanjeev Khanna, and Tan Wang-Chiew. Why and where: A characterization of data provenance. In Jan Van den Buschhe and Victor Vianu, editors, *Proceedings of the Eighth International Conference on Database Theory*, volume 1973 of *Lecture Notes in Computer Science*, pages 316–330, Berlin, Germany, January 2001. Springer-Verlag.

[7] Mark Dredze. *Intelligent Email: Aiding Users with AI*. PhD thesis, Computer and Information Science, University of Pennsylvania, 2009.

[8] Mark Dredze, John Blitzer, and Fernando Pereira. "sorry, i forgot the attachment:" email attachment prediction. In *Proceedings of the Third Conference on Email and Anti-Spam*, July 2006.

[9] Mario Frank, Joachim M Buhman, and David Basin. Role mining with probabilistic models. *ACM Transactions on Information and System Security (TISSEC)*, 15(4):15, 2013.

[10] Carrie Gates and Carol Taylor. Challenging the anomaly detection paradigm: a provocative discussion. In *Proceedings of the 2006 workshop on New security paradigms*, pages 21–29. ACM, 2006.

[11] Mieke Jans, Benoît Depaire, and Koen Vanhoof. Does process mining add to internal auditing? an experience report. In *Enterprise, Business-Process and Information Systems Modeling*, pages 31–45. Springer, 2011.

[12] William Jones and Harry Bruce. A report on the nsf-sponsored workshop on personal information management, seattle, wa, 2005. In *Report on the NSY PIM Workshop, January*, pages 27–29, 2007.

[13] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In Jean-François Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *Proceedings of the 15th European Conference on Machine Learning*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226, Berlin, Germany, September 2004. Springer-Verlag.

[14] Martin Kuhlmann, Dalia Shohat, and Gerhard Schimpf. Role mining - revealing business roles for security administration using data mining technology. In *Proceedings of the eighth ACM symposium on Access control models and technologies*, SACMAT '03, pages 179–186, New York, NY, USA, 2003. ACM.

[15] Andrew McCallum, Xueri Wang, and Andres Corrada-Emmanual. Topic and role discovery in social networks with experiments on enron and academic email. *Journal of Artificial Intelligence*, 30:249–272, 2007.

[16] Bethany McLean and Peter Elkind. *The Smartest Guys in the Room: The Amazing Rise and Scandalous Fall of Enron*. Penguin Group, New York, NY, USA, 2003.

[17] Aubrey J Rembert. Comprehensive workflow mining. In *Proceedings of the 44th annual Southeast regional conference*, pages 222–227. ACM, 2006.

[18] Jürgen Schlegelmilch and Ulrike Steffens. Role mining with orca. In *Proceedings of the tenth ACM symposium on Access control models and technologies*, pages 168–176. ACM, 2005.

[19] Asaf Shabtai, Yuval Elovici, and Lior Rokach. *A Survey of Data Leakage Detection and Prevention*

Solutions. Springer Briefs in Computer Science. Springer, New York, NY, USA, 2012.

[20] Jitesh Shetty and Jafar Adabi. Enron dataset ex employee status report.

[21] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, pages 305–316, May 2010.

[22] Wil MP van der Aalst and Ana Karla A de Medeiros. Process mining and security: Detecting anomalous process executions and checking process conformance. *Electronic Notes in Theoretical Computer Science*, 121:3–21, 2005.

[23] Wil MP Van der Aalst and Wil van der Aalst. *Process mining: discovery, conformance and enhancement of business processes*. Springer, 2011.

[24] Wil MP van der Aalst, Boudewijn F van Dongen, Joachim Herbst, Laura Maruster, Guido Schimm, and AJMM Weijters. Workflow mining: a survey of issues and approaches. *Data & knowledge engineering*, 47(2):237–267, 2003.

[25] W.M.P. van der Aalst, H.A. Reijers, A.J.M.M. Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, and H.M.W. Verbeek. Business process mining: An industrial application. *Information Systems*, 32(5):713 – 732, 2007.