Towards Mechanisms for Detection and Prevention of Data Exfiltration by Insiders

Keynote Talk Paper

Elisa Bertino
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907, USA
bertino@purdue.edu

Gabriel Ghinita
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907, USA
gghinita@purdue.edu

ABSTRACT

Data represent an extremely important asset for any organization. Confidential data such as military secrets or intellectual property must never be disclosed outside the organization. Therefore, one of the most severe threats in the case of cyber-insider attacks is the loss of confidential data due to *exfiltration*. A malicious insider who has the proper credentials to access the organization databases may, over time, send data outside the organization network through a variety of channels, such as email, crafted HTTP requests that encapsulate data, etc. Existing security tools for detection of cyber-attacks focus on protecting the boundary between the organization and the outside world. Numerous network-level intrusion detection systems (IDS) exist, which monitor the traffic pattern and attempt to infer anomalous behavior. While such tools may be effective in protecting against external attacks, they are less suitable when the data exfiltration is performed by an insider who has the proper credentials and authorization to access resources within the organization. In this paper, we argue that DBMS-layer detection and prevention systems are the best alternative to defend against data exfiltration because: (1) DBMS access is performed through a standard, unique language (SQL) with well-understood semantics; (2) monitoring the potential disclosure of confidential data is more effective if done as close as possible to the data source; and (3) the DBMS layer already has in place a thorough mechanism for enforcing access control based on subject credentials. By analyzing the pattern of interaction between subjects and the DBMS, it is possible to detect anomalous activity that is indicative of early signs of exfiltration. In the paper, we outline a taxonomy of cyber-insider dimensions of activities that are indicative of data exfiltration, and we discuss a high-level architecture and mechanisms for early detection of exfiltration by insiders. We also outline a virtualization-based mechanism that prevents insiders from exfiltrating data, even in the case when they manage to gain control over the network. The protection mechanism relies on explicit authorization of data transfers that cross the organizational boundary.

ASIACCS '11, March 22–24, 2011, Hong Kong, China.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection – *Access controls, Authentication, Cryptographic controls, Information flow controls, Invasive software (e.g., viruses, worms, Trojan horses)*

General Terms

Management, Design, Security.

Keywords

Insider Threat, Data Exfiltration.

1. INTRODUCTION

Organizations ranging from government institutions (e.g., military, judiciary, etc.) and contractors to commercial enterprises, research labs, etc., are witnessing an increasing amount of sophisticated insider attacks that are difficult to mitigate with existing security mechanisms and controls. Insider threats are staged either by disgruntled employees, or by employees engaged in malicious activities such as espionage. One of the most important objectives of insiders is to exfiltrate sensitive data such as military plans, trade secrets, intellectual property, etc.

The effectiveness of insider attacks is often higher than conventional (external) attacks for a number of reasons. First, insiders already possess credentials that allow them legitimate access to machines and services inside the organization network. Second, the actions of insiders originate at a trusted domain within the network, and are not subjected to thorough security controls in the same way as external accesses are. For instance, within the organization network there is often no internal firewall, allowing insiders to stage a broader range of attacks. Third, insiders are often highly-trained computer technicians, who have good knowledge about the internal configuration of the network and the security and auditing controls being deployed. Therefore, they may be able to circumvent conventional security mechanisms. Finally, insiders may have physical access to organization machines. This could give them the possibility to perform a much broader range of actions than those available to remote attackers.

Data exfiltration is one of the most serious threats posed by malicious insiders. Leakage of confidential data about military plans or business strategies has dire consequences. Note that, detecting exfiltration by insiders is a difficult task, and one that

cannot be successfully performed with security controls that are designed for external attacks. One of the reasons why conventional firewalls/anomaly detection systems (ADS) are not able to defend against data exfiltration is that insiders may choose among many available venues to transfer data beyond organizational boundaries. Often, exfiltrated data are piggybacked on top of conventional traffic, such as web, email or instant messaging. A study published in [2] measures the breakdown of attacks involving data exfiltration with respect to the network protocols and services used for transport. More than twelve different methods of exfiltration are identified, including outgoing HTTP requests, IRC channels, anonymous FTP, Microsoft Windows network shares, SMTP, etc. Note that, due to the flexibility available to attackers, traditional filtering/blocking approaches are not feasible. If attackers use the same channel of communication that is used by legitimate applications, then filtering will block non-malicious traffic as well. Deep-packet inspection is also not suitable, as insiders often use encoding or encryption before shipping data off-site. Thus, existing network or operating system level inspection tools that check for anomalous activity are easily circumvented.

On the other hand, data are often stored in a DBMS which enforces access control to ensure that only authorized subjects are able to access data. However, access control does not solve the problem of malicious insiders who have the proper credentials to read data. Nevertheless, by analyzing the pattern of interaction between subjects and the DBMS, it is possible to detect anomalous activity that is indicative of early signs of exfiltration. An ADS that functions at the DBMS layer (i.e., at the data source) is a promising approach towards detection of data exfiltration by malicious insiders for the following reasons: **(1)** DBMS access is performed through a standard, unique language (SQL) with well-understood semantics. It is therefore feasible to baseline behavior at this layer, as opposed to doing so at the network or operating system layer, where the diversity of mechanisms and protocols for data transfer creates complexity that often confuses conventional anomaly detection tools. **(2)** Monitoring the potential disclosure of confidential data is more effective if done as close as possible to the data source. Therefore, the DBMS layer is the most suitable place for detection of early signs of data exfiltration. **(3)** The DBMS layer already has in place a thorough mechanism for enforcing access control based on subject credentials. Additional information about the subject requesting the data, such as role, clearance, etc., is instrumental in detection of early signs of exfiltration. However, such fine-grained information is not available at other layers of the organization network (e.g., the network or OS layer).

In this paper, we discuss approaches for detecting and protecting against data exfiltration from a DBMS by insiders. We argue that protection should be achieved through careful monitoring of activity at the DBMS layer. Specifically, we identify four dimensions of tasks and actions executed by cyber-insiders during data exfiltration missions; these are: identification of data sources, data retrieval from the DBMS, lateral movement and exfiltration proper, i.e., transferring data outside the organization. We investigate typical actions for each of these four dimensions, and we discuss possible fine-tuned mechanisms for early detection of data exfiltration by insiders, in order to minimize the rate of false positives.

We will also identify factors that characterize the behavior of subjects when interacting with the DBMS. Based on these factors, it is possible to build usage profiles and identify observable DBMS activities that may be indicative of ongoing data exfiltration. The next step is to devise expected usage baselines and define threshold characteristics for each activity that will result in flags signaling potential data exfiltration. The main idea is to identify sequences and combinations of actions that are characteristics of data exfiltration. Such actions are permanently recorded in a repository of events. Subsequently, detection mechanisms are executed that take as input the individual events and determine with high accuracy whether an exfiltration mission is in progress. We will also briefly touch upon strategies for deciding when to raise alarms based on existing flags, such that detection accuracy is high (i.e., eliminate false positives). In addition, we will discuss the necessity of strategies that correlate warning flags at the DBMS layer with flags provided by other security information management (SIM) tools, network and OS anomaly detection tools, etc., to improve the accuracy of malicious insider detection.

The rest of the paper is organized as follows: in Section 2, we identify a number of activities that are indicative of data exfiltration by insiders, grouped along four distinct dimensions. In Section 3, we outline the characteristics of a pattern matching based mechanism to create profiles of nominal user behavior and to detect anomalous behavior with respect to DBMS accesses. In Section 4, we discuss mechanisms for prevention of data exfiltration based on explicit authorization of data transfers enforced at the organizational boundaries. We briefly survey related work in Section 5 and we conclude with directions for future work in Section 6.

2. IDENTIFICATION OF CYBER-INSIDER MISSION ACTIVITIES, TASKS AND ACTIONS

As the first step towards building accurate mechanisms for detection of ongoing data exfiltration, it is important to identify the individual events that may represent signs of malicious cyber-insider actions. This step aims at characterizing data exfiltration behavior, and must address fundamental questions such as:

- "*What are the distinguishing characteristics of data queries with the purpose of exfiltration?*"
- "*Which data sources does an insider target?*"
- "*What information should be collected to detect such actions?*"

In the rest of this section, we will explore in detail each of these questions.

We envision an approach to data exfiltration detection which builds upon the assumption that exfiltration represents an anomalous state that can be distinguished from the legitimate actions executed in a DBMS. Therefore, a central component of the anomaly detection framework for DBMS consists of building accurate DBMS access profiles. Such profiles can be created at various levels of granularity, i.e., one profile for all database subjects, or customized profiles for each particular role. In our presentation, we will focus on building profiles for each database role. Earlier work in [7] showed that role-based profiling is a feasible approach that allows for accurate characterization of

Table 1. Summary of Dimensions and Actions within Cyber-Insider Exfiltration Mission

Dimension of Activity	Action
(A) Identify data sources	1. Learn DBMS schema
	2. Find objects that contain sensitive data
	3. Learn authorization settings for tables, views, columns, and rows.
	4. Issue fake interrogations to conceal tracks of exfiltration
(B) Retrieve data from DBMS	1. Transfer preparation (i.e., *narrow-down*) queries
	2. Bulk transfer queries
	3. Issue decoy queries
(C) Lateral movement	1. Transfer of results within organization
	2. Changes to authorization permissions
	3. Hide/deactivate staging resources
	4. Encryption of data
(D) Exfiltration (proper)	1. Transfer of data outside organization
	2. Hide data transfer within complex operations
	3. Initiate fake transfers to check for surveillance signs

profiles. Next, we enumerate some factors that must be considered when building such profiles, with focus on the actions that are typical of an insider exfiltration mission. Note that, some of these actions occur individually within normal operations of DBMS. Therefore, our aim is to provide a thorough characterization and classification of suspicious actions, which will be used later on in the detection phase (described in Section 3) to accurately separate true exfiltration activity from false positives.

Table 1 presents a summary of the dimensions that make up the insider exfiltration mission, and the corresponding activities. Dimensions *A* and *B* are strictly related to DBMS operation and the SQL query language. Dimensions *C* and *D* are concerned with the correlation among DBMS events and events captured by other tools such as network ADS, service logs and security monitors (e.g., authentication/authorization logs), etc.

Dimension A: Identifying Sources of Data

As a first action in the cyber-insider exfiltration mission, the adversary will search for sources of data that contain sensitive information. In a DBMS, data objects are represented by tables, views, rows and columns. Dimension *A* consists of discovery of such objects that contain sensitive information, as well as discovery of the authorization permissions associated to these objects. We identify four actions within this dimension:

1. Learning the DBMS schema. The cyber-insider will issue DBMS queries that reveal the structure of the database(s) existent in the organization. This action can be performed either directly, by querying the DBMS structure, or indirectly, by querying certain repositories that contain documentation, diagrams, etc., about the database structure. In the former case, the "tell" of suspicious activity is represented by the issuance of an unusual amount of DDL (data definition language) queries, such as SHOW TABLES, SHOW COLUMNS, etc. In the latter case, the sign of suspicious activity is accessing internal data from a documentation repository, which is typically clearly marked within an organization and can be easily monitored.

2. Finding objects that contain sensitive data. This is the equivalent of a "reconnaissance" step to decide which data to target. Once the schema is known, the cyber-insider will query selected parts of the data, to evaluate which objects contain sensitive information worth exfiltrating. Due to the large data size, the insider needs to carefully choose which data to exfiltrate, in order to avoid triggering alarms based on the amount of retrieved data (basic filters in use today trigger alarms when the result set size, expressed either as the number of records retrieved, or as the amount of data bytes transferred, exceeds a certain threshold).

3. Learning authorization settings for tables, views, columns and rows. The cyber-insider will not risk attempting to read protected objects that typically trigger alarms in case of unauthorized accesses. Therefore, the insider will perform a necessary step of checking the authorization settings first. Such actions will be represented by commands that manipulate access control objects, e.g., SHOW GRANTS. Recording such commands in the event log is common practice in existing DBMS, and a query workload that consists mainly of such commands may be an indicator of suspicious activity.

4. Issuing fake interrogations to conceal the tracks of the exfiltration mission. Cyber-insiders are typically skilled individuals that are aware of the security controls in place within the organization. Therefore, the insider will constantly attempt to circumvent such controls, and also to learn whether his or her activities are currently under surveillance. To that extent, the insider will issue fake queries in the attempt to convey legitimacy to the activities executed from categories A1-A3 above. "Decoy" queries that do not provide useful information, or repetitive innocuous queries, e.g.,

```
SELECT to_char(sysdate, 'DD-Mon-YYYY
HH24:MI:SS') FROM dual;
```

may be inserted on purpose to deceive alarms. Taking into consideration such queries may give indication of suspicious activity.

Dimension B: Retrieving Data from DBMS

Once the data sources have been identified, the second dimension of activities executed by the cyber-insider is concerned with retrieving the actual data from the database. Note that, this does

not necessarily imply transfer of data outside the organization (this aspect is addressed in Dimension *D*), as cyber-insiders typically use staging servers within the same administrative domain to cache data before exfiltrating them outside the organizational boundaries.

1. Transfer Preparation. To avoid large-sized data transfers, the cyber-insider will attempt to narrow down the information that is being retrieved, in order to keep the result count low (and hence not trigger alarms). Certain queries may exhibit a clear pattern of *"narrowing-down"* in order to restrict query result size. For instance, consider that the insider wants to issue the query

```
SELECT * FROM employees WHERE proj_name =
"defense_project";
```

Since many employees may be involved in the project, the insider will first issue a COUNT query that returns the *number* of results, but not the actual results. Such "*guard*" queries, which are not typically occurring within normal system operation may be an important sign of suspicious activity.

2. Bulk Transfer Queries. The insider executes SQL queries to retrieve confidential information from the database. Such queries are likely to return a large number of results compared to typical queries. Therefore, a sign of ongoing exfiltration is that queries have *low selectivity*. The insider will attempt to break down such queries into ones with higher selectivity, but whose combined results still give the desired information.

3. Issue decoy queries. To cover his or her tracks, the cyber-insider will issue additional decoy queries that do not provide any benefit with respect to the exfiltration mission, but instead have the purpose of concealing malicious activity and confusing anomaly detection tools. For instance, to create queries with high selectivity (and therefore increase the amount of query selectivity on average), query predicates that are certain to return no results may be issued. For instance,

```
SELECT * WHERE attribute_name !=
attribute_name;
```

will always return zero results, circumventing certain syntactic query checkers that may look for patterns of data retrieval "en-masse" (i.e., queries without selection predicates).

Dimension C: Lateral Movement

In order to prevent detection, cyber-insiders often engage in a number of activities aimed at concealing the attack tracks, as well as at learning whether the activities of the attacker are being monitored. To confuse the security controls, the cyber-insider may perform the following activities:

1. Transfer of results within the organization. The insider initiates a data transfer from the database client machine to a staging server. This operation may use non-standard data transfer protocols, and typically involves transferring large amounts of data.

2. Changes of authorization rights and resource permissions. With the aim of hiding its malicious actions, the insider will restrict the access to the staged data, such that other organization members with access to the machine will not be able to detect the presence of the staged data. Signs of such actions are the use of OS or database-level access control configuration tools and wizards. Note that, conventional auditing mechanisms should log such special commands, in order to facilitate accurate detection, as described in Section 3.

3. Hide and/or deactivate resources used for staging. The insider may temporarily take a machine off the network, or disable services such as remote access (for other users) to cover his/her tracks. For instance, the insider may want to prevent other users from being able to notice a decrease in available disk space on the staging server. A sign of such activity occurring may be represented by unexpected lack of resource availability. Similar to case A3, such events should be captured and logged for future analysis in the process of exfiltration detection.

4. Encryption of data. The cyber-insider will attempt to prevent other users or system administrators from finding that confidential data is stored outside the allowed security realm of the DBMS. For instance, routine file system scanners may scan files for certain patterns that are indicative of confidential data. To cover its tracks, the malicious cyber-insider may encrypt data to prevent exposure of staged data. Encryption typically requires considerable amounts of computation; therefore signs of unusually high CPU utilization may be indicative that some potentially malicious anomalous activity is ongoing.

Dimension D: Exfiltration (proper)

This dimension is concerned with activities of transferring sensitive data across the organizational boundaries.

1. Transfer of data outside the organization. The cyber-insider will initiate a number of data transfers, possibly disguised in the form of innocuous protocols such as HTTP. The sign of this step may be the presence of network flows to a host that is not typically part of the set of destination hosts for outside connections.

2. Hide data transfer within complex operations. Rather than issuing a stand-alone exfiltration transfer which may trigger alarms, the insider may execute a series of legitimate transfers in sequence, except for one single session that sends the sensitive data. However, seen as a whole, the entire batch of transfers may evade detection.

3. Initiate fake transfers to check for surveillance signs. Before exfiltrating the actual data, the insider may attempt a series of transfers that are innocuous, but similar in nature to the characteristics of the planned exfiltration step (e.g., similar destination IP address, similar amount of bytes transferred, etc). If an alarm is raised for the fake transfer, the insider will learn that such an approach will be detected. Furthermore, the insider will have a good motivation for the innocuous transfer, so his/her identity will not be placed on a watch list if an alarm is triggered (instead, the event will be flagged as a false alarm).

3. ANOMALOUS EVENT CORRELATION AND ANALYSIS

In this section, we discuss a high-level architecture of a DBMS-centric mechanism for detecting data exfiltration by malicious insiders. The key idea underlying this approach is to identify actions, tasks and sequences thereof, that are part of the exfiltration mission carried out by a cyber-insider. To distinguish cyber-insider mission actions from legitimate user activities, it is necessary to build profiles of nominal behavior for each role present in the DBMS. These role profiles are subsequently used to detect anomalous behavior that is indicative of exfiltration. Note that, DBMS role access profiles fit naturally in typical deployment contexts, since the database already has a Role Based Access Control (RBAC) mechanism in place. Authorizations are specified with respect to roles and not with respect to individual users. One or more roles are assigned to each user and privileges are assigned to roles. The system that we discuss is assumed to build a profile for each role and will be able to determine role intruders, that is, individuals that while holding a specific role deviate from the normal behavior of that role.

We identify three main tasks that this architecture must provide, as follows:

(1) Identify the individual actions pertaining to DBMS accesses that are part of the cyber-insider mission to exfiltrate sensitive information from within the organization. All such events must be recorded in an audit log for further processing in subsequent steps.

(2) Determine what are the inter-relationships and correlations between individual actions and dimensions of activities, and devise mechanisms that are capable of recognizing and reconstructing cyber-insider exfiltration missions by assembling together individual actions from the event log. One objective of this step is to maximize detection accuracy, i.e., minimize the rate of false positives.

(3) Cross-check the sequence of tasks and activities within the recognized cyber-insider mission with other sets of events and red flags raised at other monitoring layers, such as operating system and network layer ADSs. Even though by themselves these additional signals are not sufficient for effective exfiltration detection, they are used to confirm the threats identified at the DBMS layer and thus to increase the accuracy of the overall mechanism.

Figure 1 shows the architecture of the proposed exfiltration detection mechanism. The flow of interactions between the modules involved in exfiltration detection is as follows: during the training phase, the SQL commands submitted to the DBMS (or read from the audit log) are analyzed by the profile creator module to create the initial profiles of the database users. For every SQL command under detection, the feature selector module extracts the features from the queries in the format expected by the detection engine. The detection engine then runs the extracted features through the detection algorithm. If an anomaly is detected, the detection mechanism submits its assessment of the SQL command to the response engine according to a pre-defined interface; otherwise the command information is sent to the profile creator process for updating the profiles. Note that, the fact that a query is anomalous may not necessarily imply ongoing exfiltration. Other information such as network ADS and security policies must also be taken into account. For example, if the user logged under the role is performing some special activities to manage an emergency, the detection mechanism may be instructed not to raise alarms in such circumstances.

From a system design point of view, Figure 2 shows the various components involved in event processing and correlation, as well as the interaction between them. The DBMS records the access events into the raw event log, which represents the input for the analysis tasks. However, the raw event log is not suitable for direct processing, since the data are not properly organized and indexed to allow for fast response to a large number of exfiltration detection strategies. Therefore, an index of events is created, which contains sequences and combinations of events that can be efficiently interrogated by the analysis modules. Furthermore, additional metadata such as scores that measure the confidence in correlation between certain sequences of events and the presence of exfiltration are maintained and updated during analysis.

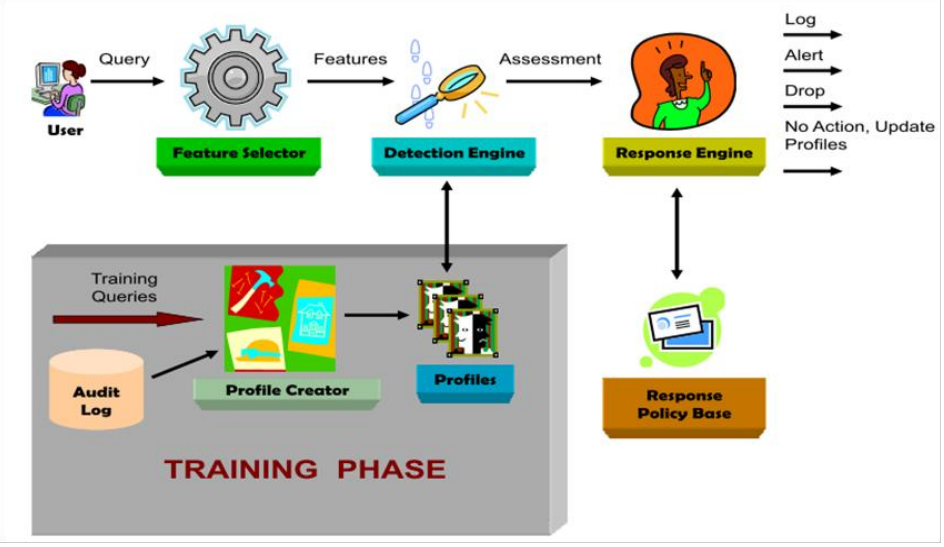

Figure 1: Architecture of DBMS-level Exfiltration Detection Mechanism

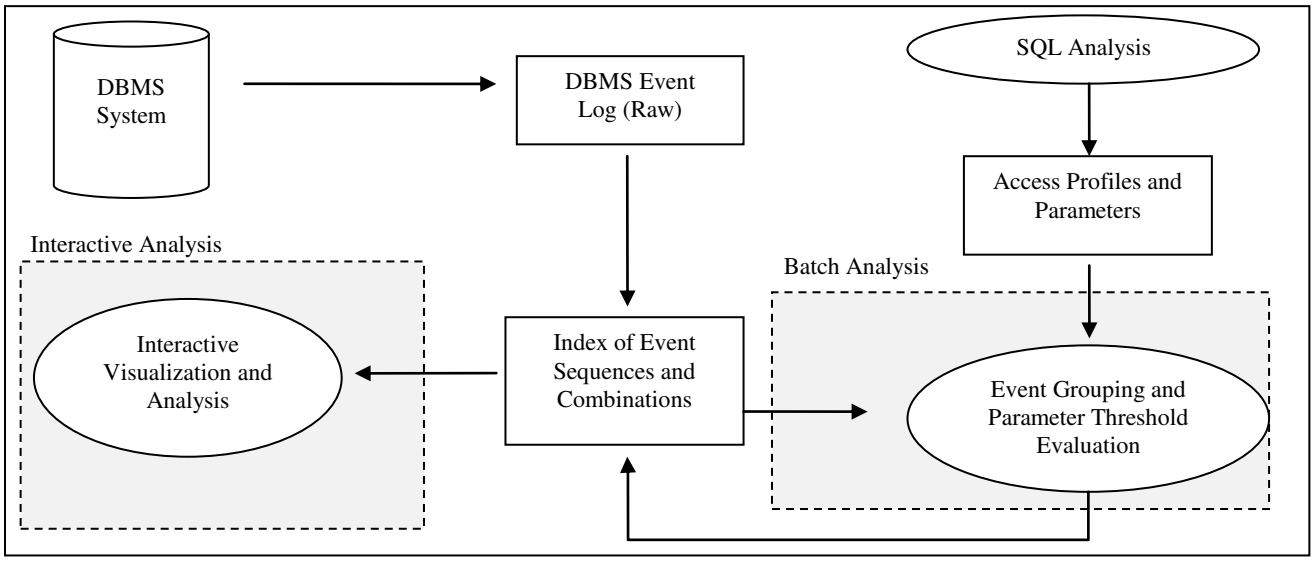

Figure 2: Framework for Event Processing and Exfiltration Mission Detection

As part of this process, it is important to set up and maintain an efficient structure for the event log repository, as well as to devise fast techniques for indexing and querying the information related to DBMS events. In addition, it is important to investigate the aspects related to event correlation analysis and mechanisms for raising alarm flags.

The SQL feature analysis component is an important part of the model, and consists of a thorough characterization of SQL commands with respect to a number of parameters that are relevant to the action list identified in Section 2. In developing this component, it is important to consider criteria such as query type (e.g., DDL-data definition language or DML-data manipulation language), query result set count, query selectivity, number of tables involved in a query, etc. Another important aspect to investigate is the use of methods for detection of query *equivalence*. This aspect is important to decide whether the cyber-insider is decomposing a large data transfer into a larger number of individual queries, in order to avoid detection. This is likely to be executed in action *2-bulk transfer* of dimension *B – Retrieve data from DBMS* (please see Table 1). Therefore, identification and observation of queries correlated over time (although issued in separate sessions to avoid detection) is very important.

As a result of the SQL analysis, a number of parameters and criteria, as well as associated thresholds will be obtained and fed as input to the analysis phase. The objective of the analysis is to evaluate with the help of a training dataset to what extent certain parameters, parameter value ranges and combinations thereof are highly accurate in detecting exfiltration. For instance, query selectivity may not necessarily be a clear indicator of exfiltration by itself. If a query retrieves all the tuples in a table, it may be an indication that an insider is trying to exfiltrate large amounts of data, but it may also mean that some legitimate backup is being performed to another system. Additional information such as the role of the user executing the query may be needed to make the correct decision of raising an alarm or not.

We envision two types of event information analysis that should be performed to achieve accurate detection, each of them supported by a distinct module:

- **Batch Analysis.** This module executes search heuristics in the space of user actions and parameter values in order to find combinations of actions and settings of parameter thresholds that yield high accuracy for detection. Note that, there is a feedback loop between the batch analysis module and the index of events: as the batch analysis module evaluates the detection accuracy for various event groupings and parameter threshold settings, the output of the evaluation is reflected in the score assignment within the index structure.

The other input to the batch analysis module is the set of access profiles and associated parameters that are obtained as an outcome of the SQL features obtained in the SQL analysis of actions identified in Section 2. The access profiles are used to guide the heuristic search for tuning the detection mechanisms, and the initial parameter settings from the SQL analysis step are continuously updated to reflect the settings corresponding to the most accurate combinations found so far.

- **Interactive Analysis.** Often, it may not be feasible or efficient to perform batch analysis in order to identify trends and correlations in detection accuracy over a large space of actions and parameters. This issue can be addressed by creating a tool for interactive visualization of detection accuracy over sub-sets of actions and parameters. This step will allow a human operator to better understand the correlations among actions and create efficient filters that increase detection accuracy. Such a tool would be helpful to better understand the characteristic features of exfiltration missions in general, and may be used as a standalone product as well.

4. MECHANISMS FOR PREVENTION OF DATA EXFILTRATION

So far, we have discussed an architecture for detection of data exfiltration based on anomaly detection at the level of DBMS activities. Data mining and pattern-matching were employed to create baseline usage profiles and to determine whether an

ongoing activity shows signs of malicious behavior or not. However, such an approach is statistical in nature, and cannot deterministically detect without false positives and negatives whether exfiltration takes place. In this section, we look at two alternative approaches that aim at preventing data exfiltration. First, in Section 4.1 we discuss a technique for embedding provenance within data in the form of difficult-to-remove watermarks that are undetectable by attackers. Confidential data are marked appropriately and border gateways can check whether data are allowed to leave the organization. Second, in Section 4.2 we overview a virtualization-based architecture that creates separate, isolated segments of the organization network, namely one trusted and one non-trusted segment. Data that attempt to pass from the trusted segment to the non-trusted one (which includes outside networks and the Internet) are required to have explicit credentials that are checked by an authorization gateway situated at the border of the trusted network. Any data that are not explicitly authorized are not allowed to leave the trusted domain, hence exfiltration is not possible. To address cyber-insider threats, a threshold-based credential generation mechanism can be employed, whereby a number of independent principals are required for a transfer to be permitted.

4.1 Tracking Lineage with Watermark-based Provenance Embedding Techniques

One important aspect in detecting malicious insiders is to track closely their activities. Earlier in Section 3, we have discussed the importance of monitoring and logging network events. However, tracking individual, disparate events may not always give a clear indication of malicious activity. In addition, in order to correlate multiple events, it is often required to track data items, and to know what were the actions and tasks that lead to the generation of those data items.

Consider that an insider plants malware in one of the organization machines. In turn, the malware will affect certain files, either locally on the same machine, or remotely on other machines on the organization network. Since insiders may try to cover their tracks before staging an attack, the malware may travel across several hosts before starting to execute the attack proper. Provenance and lineage techniques can help in the early identification of malware propagation, and trigger alarms that some malicious activity may be in progress. Thus, secure collection and processing of provenance information can help greatly in defending against data exfiltration attacks.

Tracking data and event lineage may be achieved using additional tags that can be attached at various layers of abstraction (e.g., inside each network packet). On the other hand, such explicit tags may be visible to the attacker, and alert the insider that his or her actions are being monitored. The cyber-insider may thus react by either delaying the attack, or attempting to circumvent the provenance tracking mechanism. Therefore, it is desirable to devise tracking mechanisms that are transparent and resilient to removal, in the form of watermarks.

A watermark embedded in the network packets can serve the purpose of tracking malicious activities in a stealthy fashion. Furthermore, unauthorized possession of data by malicious insiders may also be achieved with watermarks. Consider that an insider gathers sensitive data from within the organization and stores them on a compromised staging machine or server with the purpose of exfiltrating them at a later time. If data are watermarked, then a simple scanning software function can detect the presence of such data (e.g., a plug-in for an anti-virus tool may achieve this functionality). Possession of unauthorized data will signal that an insider attack is in progress.

The embedded provenance information can serve as a criterion of filtering data transfers at the organizational boundary gateway. For instance, a trusted border router that knows the secret transformation and/or transformation keys used to embed provenance can check the lineage information within the data and determine whether any confidential data were at the origin of the generation of the current packet that is scheduled to leave the organization. If the presence of the confidential data is detected, then the packet will be dropped and an alarm will be raised indicating attempted exfiltration by a malicious cyber-insider. The provenance data can further be used for forensics purposes, to narrow down the identity and/or source in the network where the confidential data leak originated.

4.2 A Virtualization-based Mechanism for Prevention of Data Exfiltration

Most often, getting access to important confidential data is the main motivation of insiders who are spending considerable amounts of time and resources to plan and execute sophisticated attacks against carefully selected targets. Even if attackers are able to compromise certain parts of the organizational network, the damage inflicted may not be significant, unless critical information about the organization is leaked. Such information may include military secrets, specifications of proprietary technologies, confidential business strategy details, etc.

In practice, full protection against cyber-insider attacks may be an unfeasible goal, due to the complexity of organization-level networks that often include a large variety of applications, protocols, services, etc. This heterogeneity provides adversaries with a broad spectrum of attack vectors, which complicates the task of defending such networks. Therefore, it is important to address the outcome when some of these attacks may succeed, and in these scenarios it is essential to limit the ability to exfiltrate confidential data. In the following, we will discuss an effective mechanism that prevents such data leakage.

Method	Frequency
Native Remote Access Applications	27%
Microsoft Windows Network Shares	28%
Malware Capability: FTP	17%
Malware Capability: IRC	2%
Malware Capability: SMTP	4%
HTTP File Upload Site	1.5%
Native FTP Client	10%
SQL Injection	6%
Encrypted Backdoor	<1%

Figure 3: Breakdown of Data Exfiltration Attacks per Transport Mechanism Used

One of the reasons why conventional firewalls and ADSs are not able to defend against data exfiltration is the fact that sophisticated cyber-insiders may choose among many available venues to transfer data beyond organizational boundaries. Often, exfiltrated data are piggybacked on top of conventional traffic, such as web, email or instant messaging. Figure 3 shows the result of a study [2] that measures the breakdown of attacks involving

data exfiltration with respect to the network protocols and services used for transport. Note that, due to the flexibility available to attackers, traditional filtering/blocking approaches are not feasible. If attackers use the same channel of communication that is used by legitimate applications, then filtering will block non-malicious traffic as well. Deep-packet inspection is also not suitable, as attackers often use encoding or encryption before shipping data off-site. Thus, packet inspection tools that check for tokens that may indicate sensitive packet contents are easily circumvented.

Rather than using general-purpose filters, a more effective approach is a mechanism based on credentials associated with the data. Data items that are sensitive are marked correspondingly, and require explicit authorization to depart the organization network. Specifically, we envision a *"confine-and-mark"* approach for protection against exfiltration of sensitive data. The architecture of this approach is presented in Figure 4. The confine-and-mark approach tackles the exfiltration problem from two angles:

1. Restrict the segment(s) of the network from which sensitive data can be accessed
2. Label sensitive data with cryptographic authorization tokens, such that only data that have been approved for transfer can leave the protected network

In summary, the two steps in the confine-and-mark approach work as follows:

- **Confine Step.** Confinement of sensitive data is achieved through a combination of network segmentation and virtualization. Segmentation physically prevents sensitive data from being sent to certain machines with low levels of trust, whereas virtualization employs trusted virtual machine monitors (VMM) that restrict access to sensitive data only to trusted applications. The enforcement of separation within the same physical machine can be done using encryption, e.g., the Overshadow-style [3] approach.
- **Mark step.** Sensitive information is marked accordingly, and the markup is transferred from one type of data representation to another. For instance, if a file on disk is marked confidential, all data packets that are generated as a result of transferring the file must also contain the proper label, and must be given special authorization credentials to leave the network. It is important to investigate authorization policy semantics and enforcement for access to sensitive data. To control the flow of sensitive data, cryptographic tokens should be used, such that the integrity and confidentiality of the markup can be ensured. To enforce transfer authorization policies, it is necessary to develop mechanisms based on application-level *authorization gateways* that will filter all traffic originating at trusted machines/network segments. Therefore, a trade-off emerges between the size of the trusted segment and performance: the more trusted machines, the more costly the transfer authorization step. On the other hand, if the sensitive data flow is restricted to few machines, most (non-sensitive) traffic will not need to be routed through the gateway. The authorization checks in this model can be enforced at lower OSI stack layers as well, not only at the application layer. For instance, this can be done by labeling and checking authorization for individual network packets, or at the level of user-initiated sessions.

The confine-and-mark approach raises several challenging research problem in multiple areas, including formal modeling, cryptography, networking and systems:

- **Modeling and Specification of Lineage Information.** The set of sensitive data items is not static. Instead, it evolves dynamically: every time computations are performed on the data, the results are also sensitive, and need to be protected. Tracking the data flow and ensuring that sensitive data items are properly marked is not a trivial task. In addition, a challenging goal is to devise mechanisms that not only preserve the lineage of the data, but also automatically extend the cryptographic authentication and authorization tokens to derived results. This way, there is no need to individually authorize derived data items, improving system usability and

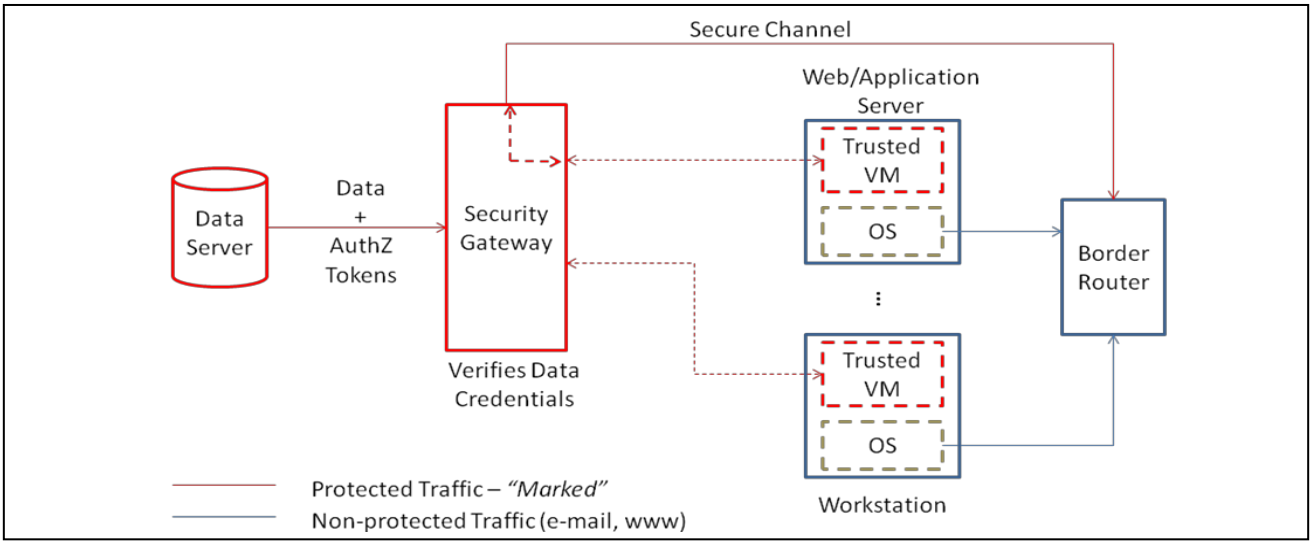

Figure 4: "*Mark-and-Confine*" Approach to Protect against Data Exfiltration

increasing transfer throughput. To ensure correctness, it is important to derive formal models for automatic extension of cryptographic tokens to derived data items.

- **Authentication and Authorization.** Each transfer of sensitive data must be authorized by a subject (either an individual or a system service) whose identity needs to be verified. Therefore, secure authentication and authorization must be implemented at the security gateway level. Another important aspect is that of continuous mechanisms for authentication and authorization. To ensure that attackers do not attempt to re-use transfer credentials, the security gateway may choose to request a re-authentication of transfer after a certain timeout expires. This can be repeated per-data-flow, or even within the same flow, when security requirements are high.

Note that, since the main threat considered is that of attacks originating at cyber-insiders, it is important to take into account the situation where an individual that possesses the proper credentials attempts to exfiltrate confidential data (e.g., a high ranking executive, or a system administrator with full access to the database). To address such cases, it is important to devise threshold-based mechanisms for authorization, whereby a number of distinct principals (and their associated credentials) are required to authorize a transfer. This way, no one single principal can perform an attack.

Another important aspect is that of cryptographic key management. Since our objective is to protect against sophisticated cyber-insiders, and we assume that certain security breaches may occur, storing cryptographic keys within the reach of operating systems is not a feasible approach. Instead, it may be necessary to investigate secure key management techniques based on hardware tokens. Even if cyber-insiders gain complete control over a machine's operating system, they can still not gain access to the encryption keys. Of course, physical protection layer controls must also be used for increased security.

- **Systems issues.** Virtual machines are a key component of the proposed protection design. Virtualization ensures that sensitive data are confined to trusted transfer paths, and that it is not possible for the cyber-insiders to exfiltrate data on alternate paths that are not subject to authorization checks. However, the separation between the trusted and the un-trusted domains must be carefully planned, and take into account issues such as in-memory protection of data, trusted storage, etc. Furthermore, the use of virtualization as well as the application gateway component results in additional overhead that may considerably impact performance. Therefore, performance optimization will play a key role in making the confine-and-mark approach suitable for real-life scenarios.

- **Network issues.** The embedding of authorization tokens in network packets must be done in a transparent fashion, such that the impact on the existing network and routing infrastructure is not significant. Most network devices do not support any authentication or authorization mechanisms, therefore the design of protocols to handle modified packets must be done to allow transparent handling of embedded security information.

5. RELATED WORK

Insider threat has been widely acknowledged [1, 7, 11, 12] as a very serious and difficult to address cyber-security concern. There are several factors that make insider threats very effective, such as knowledge of the organization network, possession of valid credentials, and the benefit of trust. Data exfiltration has been recently recognized as an important research problem, and studied in several contexts. In [4], it is shown how data can be exfiltrated by embedding sensitive information in conventional browser HTTP requests. In [5], the authors study exfiltration through covert channels. For instance, the rate of sending packets to a destination can be tuned such that the frequency or inter-arrival time of packets corresponds to an encoding of the data. The work in [6] proposes a framework for detection of data exfiltration by deploying a transparent network bridge at the edge of the network. The detection technique relies on statistical and signal processing methods.

Protection of data in the presence of untrusted users or software has been addressed in [3] where a virtual machine monitor is deployed to separate sensitive data and applications from other malicious or untrusted software, e.g., malware, compromised operating system, etc. The mechanism for protection against data exfiltration discussed in Section 4 also relies on virtualization, but deals with more comprehensive challenges related to data lineage and explicit authorization of transfers.

Several approaches dealing with intrusion detection (ID) for operating systems and networks have been developed in [13-18]. However, they are not adequate for data exfiltration due to their limited ability to assess what are the contents of network packets or OS-level messages. Also, they do not work at a sufficiently fined-grained level that allows for accurate identification of exfiltration missions, and they do not focus on cyber-insiders.

An abstract and high-level architecture of a DBMS incorporating an intrusion detection (ID) component has been recently proposed in [10]. However, this work mainly focuses on discussing generic solutions rather than proposing concrete algorithmic approaches. In [16] a method for ID is described which is applicable only to real-time applications, such as a programmed stock trading that interacts with a database. The key idea pursued in this work is to exploit the real-time properties of data for performing the ID task.

Anomaly detection techniques for detecting attacks on web applications have been discussed by Vigna et al. [19]. A learning based approach to the detection of SQL attacks is proposed by Valeur et al. [20]. The motivation of that work is similar to the approach we described in Section 3. Their methodologies, however, focus on detection of attacks against back-end databases used by web-based applications. Thus, their ID architecture and algorithms are tailored for that context. Our work, on the other hand, focuses on a general purpose approach towards detection of anomalous access patterns in a database as represented by SQL queries submitted to the database.

DEMIDS is a misuse-detection system, tailored for relational database systems [8]. It uses audit log data to derive profiles describing typical patterns of accesses by database users. Essential to such an approach is the assumption that the access pattern of users typically forms a working scope which comprises sets of attributes that are usually referenced together with some values. The idea of working scopes is captured by mining frequent itemsets which are sets of features with certain values. Based on

the data structures and integrity constraints encoded in the system catalogs and the user behavior recorded in the audit logs, DEMIDS describes distance measures that capture the closeness of a set of attributes with respect to the working scopes. These distance measures are then used to guide the search for frequent itemsets in the audit logs. Misuse of data, such as tampering with the data integrity, is detected by comparing the derived profiles against the organization's security policies or new audit information gathered about the users. The goal of the DEMIDS system is two-fold. The first goal is detection of malicious insider behavior. Since a profile created by the DEMIDS system is based on frequent sets of attributes referenced by user queries, the approach is able to detect an event when a SQL query submitted by an insider does not conform to the attributes in the user profile. The second goal is to serve as a tool for security re-engineering of an organization. The profiles derived in the training stage can help to refine/verify existing security policies or create new policies. The main drawback of the approach presented in [8] is a lack of implementation and experimentation. The approach has only been described theoretically, and no empirical evidence has been presented of its performance as a detection mechanism. The approaches in [7,11,12] are currently the state-of-the-art in DBMS-layer intrusion detection, and their architecture is similar to the one discussed in Section 3.

6. CONCLUSION

Cyber-insider attacks are very effective and often highly damaging, due to their knowledge about the organization and their ability to exfiltrate confidential data. Insiders reside within the trust domain of an organization, they are not subject to the same security controls that keep external attackers at bay, and they have valid credentials to access systems and services within the organization. In this paper, we have identified dimensions of activities that are indicative of data exfiltration by insiders, and we argued that a DBMS-layer architecture for exfiltration detection is the most suitable approach to defend against this threat. We outlined mechanisms based on pattern matching that create baseline profiles for database users and detect anomalies that are indicative of malicious behavior. Finally, we have discussed a possible mechanism for prevention of exfiltration using provenance tracking and virtualization. Note that, although we have presented these approaches independently, in practice they can all be combined to achieve stronger protection from insider threats. An interesting direction for future research is to devise highly accurate detection tools and secure provenance techniques that can effectively protect against exfiltration while keeping the system overhead low, in order not to slow down legitimate data accesses and transfers.

7. REFERENCES

[1] J. Leyden, "Geeks, squatters and saboteurs threaten corporate security", http://www.theregister.co.uk/2005/12/15/mcafee_internal_security_survey/

[2] N. J. Percoco, "Data exfiltration: how data gets out", Spiderlabs report. Available online at http://www.csoonline.com/article/570813/data-exfiltration-how-data-gets-out

[3] X. Chen et al, "Overshadow: A Virtualization-Based Approach to Retrofitting Protection in Commodity Operating Systems", In Proc. Of Intl. Conf. on Architectural Support for Programming Languages and OS (ASPLOS '08)

[4] K. Born, "Browser-Based Covert Data Exfiltration", In Proceedings of the 9th Annual Security Conference, Las Vegas, NV, April 7-8, 2010

[5] A. Giani, V. H. Berk, G. V. Cybenko, "Data exfiltration and covert channels", In Proc. of Sensors, Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense, 2006

[6] Y. Liu, C. Corbett, K. Chiang, R. Archibald, B. Mukherjee, D. Ghosal, "SIDD: A Framework for Detecting Sensitive Data Exfiltration by an Insider Attack," In Proc. of Hawaii International Conference on System Sciences, Jan., 2009.

[7] A. Kamra, E. Terzi and E. Bertino, "*Detecting Anomalous Access Patterns in Relational Databases*", Very Large Data Bases Journal (VLDBJ), 2008.

[8] C. Chung, M. Gertz and K. Levitt, "*DEMIDS: a misuse detection system for database systems*", in Integrity and Internal Control in Information Systems: Strategic Views on the Need for Control, IFIP TC11 WG11.5, 2000.

[9] S. Wenhui and T. Tan, "*A novel intrusion detection model for securing web-based database systems*", In Proc. of the Annual Computer Security Applications Conference (ACSAC), 2002.

[10] P. Liu, "*Architectures for intrusion-tolerant database systems*", In Proc. of the Annual Computer Security Applications Conference (ACSAC), 2002.

[11] A. Kamra, E. Bertino: "*Privilege States Based Access Control for Fine-Grained Intrusion Response*". RAID 2010: 402-421

[12] A. Kamra, E. Bertino: "*Design and Implementation of an Intrusion Response System for Relational Databases*". IEEE Trans. on Knowledge and Data Engineering (TKDE), 2010

[13] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Tech. Rep. 99-15, Chalmers Univ., Mar. 2000.

[14] K. H. A. Hoglund and A. Sorvari, "A computer host-based user anomaly detection using the self-organizing map," in Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN), 2000.

[15] T. Lane and C. E. Brodley, "Temporal sequence learning and data reduction for anomaly detection," ACM Transactions on Information and System Security (TISSEC), vol. 2, no. 3, pp. 295–331, 1999.

[16] T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes, and T. Garvey, "A real - time intrusion detection expert system (ides)", Technical Report, Computer Science Laboratory, SRI International, 1992.

[17] R. Talpade, G. Kim, and S. Khurana, "Nomad: Traffic-based network monitoring framework for anomaly detection," in Proceedings of the 4th IEEE Symposium on Computers and Communications (ISCC), 1998.

[18] V. Lee, J. Stankovic, and S. Son, "Intrusion detection in real-time databases via time signatures," in Proceedings of the IEEE Real-Time Technology and Applications Symposium (RTAS), 2000.

[19] C. Kruegel and G. Vigna, "Anomaly detection of web-based attacks," in Proceedings of the ACM Conference on Computer and Communications Security (CCS), 2003.

[20] F. Valeur, D. Mutz, and G. Vigna, "A learning-based approach to the detection of sql attacks," in Proc. of Intl. Conference on detection of intrusions and malware, and vulnerability assessment (DIMVA), 2003