A Traceability Link Mining Approach for Identifying Insider Threats[1]

Yi Hu
Department of Computer Science
Northern Kentucky University
Highland Heights, KY 41099
Email: huy1@nku.edu

Brajendra Panda
Computer Science and Computer Engineering Department
University of Arkansas
Fayetteville, AR 72701
Email: bpanda@uark.edu

ABSTRACT

In this paper, we present an insider attack detection model that is designed to profile traceability links based on document dependencies and calendar-based file usage patterns for detecting insider threats. This model is utilized to detect insiders' malicious activities targeted at tampering the contents of files for various purposes. We apply the concept of traceability links in the software engineering field to this research. Our approach mainly employs document dependency traceability links for constructing insider attack detection model.

Categories and Subject Descriptors

K.6.5 [Management of Computing and information systems]: Security and Protection – access controls.

1. INTRODUCTION

The problem with most organizations is that employees are given a lot more access than what they actually need to do their job [3]. Although most companies pay lots of attention to outsider threats and spend significant efforts in securing information systems, very few of them adopt a systematic strategy to mitigate insider threats. In this research, we proposed two models that employ document dependency traceability links for detecting insider attacks.

2. THE MODEL

2.1 The Base Model

In this model, all files in a computer system are tagged with different categories. For example, for a project in a software repository, there are following file categories: user requirements, functional requirements, source code, documentation, software configuration management, etc. For payroll accounting systems, files can be categorized as salary, benefit, tax, insurance, etc. Inside each category, there can be many files belonging to this category.

[1] This work was supported in part by AFOSR under grant FA 9550-04-1-0429

Definition 1: Horizontal File Clique
A Horizontal File Clique H consists of related files that belong to *different* file categories in C (where the set C consists of all file categories) and these files in H are often updated together. We use H: (f_1: c_1, f_2: c_2, …, f_n: c_n) to denote a horizontal file clique, where f_i represents a file (or a set of files belonging to the same category) and c_i represents a file category, $1 \le i \le n$. Also, for $\forall H$, $\exists\, f_i$ and f_j, such that $c_i \ne c_j$, $1 \le i, j \le n$.

Definition 2: Vertical File Clique
A Vertical File Clique V consists of files that belong to the *same* file category $c_i \in C$ (where the set C consists of all file categories) and these files in V are often updated together. We use V: (f_1, $f2$, …, fn) to denote a vertical file clique, where f_i represents a file, $1 \le i \le n$.

Definition 3: Horizontal File Chain
A Horizontal File Chain $\widehat{H}$ consists of related files that belong to *different* file categories in C (where the set C consists of all file categories) and these files in $\widehat{H}$ are often updated in certain sequence. We use $\widehat{H}$: <f_1: c_1, f_2: c_2, …, f_n: c_n> to denote a horizontal file chain, where f_i represents a file (or a set of files belonging to the same category) and c_i represents a file category, $1 \le i \le n$. It specifies that if file f_1 is updated, file f_2, ..., f_n have to be updated in the sequence specified. Also, for $\forall \widehat{H}$, $\exists\, f_i$ and f_j, such that $c_i \ne c_j$, $1 \le i, j \le n$.

Definition 4: Vertical file chain
A Vertical File Chain $\widehat{V}$ consists of files that belong to the *same* file category $c_i \in C$ (where set C consists of all file categories) and these files in $\widehat{V}$ are often updated in certain sequence. We use $\widehat{V}$: < f_1, $f2$, …, fn > to denote a vertical file chain, where f_i represents a file, $1 \le i \le n$. It specifies that if file f_1 in category c_i is updated, then file f_2, …, f_n in the same category have to be updated in the sequence specified.

Suppose there is a payroll system that maintains employee files of 4 different categories, e.g., salary, benefit, tax, and insurance. At the end of the month, files from these categories are often updated together to modify each employee's record. Figure 1 illustrates two vertical file cliques and one horizontal file clique for this case. In the horizontal file clique enclosed by the outside bolded dashed lines, there are 7 files that are belonged to 4 different file categories. Inside the horizontal file clique, there are two vertical file cliques. In this example, it shows that a horizontal file clique can contain one or more vertical file cliques. One vertical file clique has 3 files belonging to *Benefit* category, say, retirement plan, stock option plan, and flexible spending account plan. Another vertical file clique has 2 files of *Tax* category, state tax and federal tax, for example.

Figure 2 presents one vertical file chain and one horizontal file chain that are reflected in the software engineering practice. A change in user requirement file causes a corresponding change in a source code file and the subsequent update in the documentation. This illustrates the relationships among three files of different categories in the horizontal file chain. The vertical file chain shows that there are 3 files that are not only related (all in the same category of *Documentation*), but have to be updated in the sequence specified.

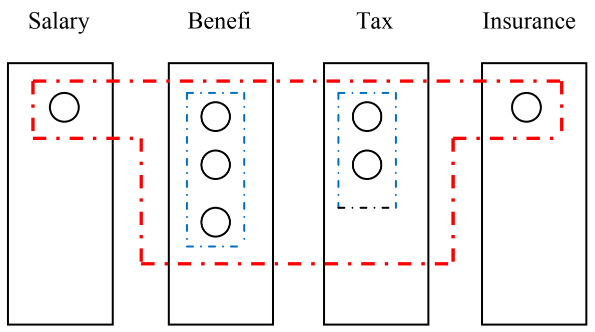

Figure 1. Vertical File Clique and Horizontal File Clique

In the base model, we designed an algorithm that employs association rule mining and sequential pattern mining for identifying horizontal file cliques, vertical file cliques, horizontal file chains, and vertical file chains based on file access logs. It must be noted that each file clique or file chain discovered is associated with certain support and confidence values. This data mining process is performed during normal system usage phase. These file cliques and file chains are used as rules for identifying malicious insider threats.

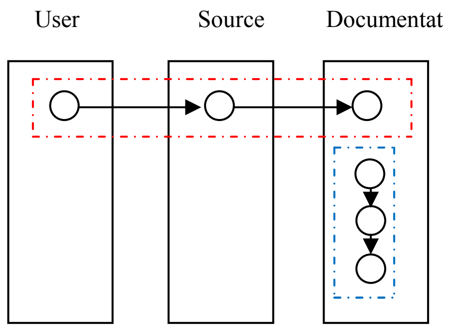

Figure 2. Vertical File Chain and Horizontal File Chain

We assume that a malicious insider has access to various files. These can be done, for example, either by using social engineering method to get access rights to these files or by masquerading as another legitimate user by stealing login credentials. However, the malicious insider does not have the knowledge of traceability links among the files, so that (s)he does not know the file dependencies specified by the file cliques and the file chains. If a malicious insider modifies certain files without updating other files related in the vertical or horizontal file cliques or without updating relevant files in correct sequence in the vertical or horizontal file chains, an alarm will be raised.

2.2 The Enhanced Model

The base Model employs *Horizontal File Clique*, *Vertical File Clique*, *Horizontal File Chain*, and *Vertical File Chain* for identifying insider threat. This model assists in identifying certain malicious insider threats but fails to catch some well-crafted malicious activities. For example, if one insider knows the particular file dependencies specified in figure 1, (s)he might be able to modify these files as required without being detected. We proposed an enhanced model that is able to catch certain insider threats of this kind by utilizing calendar-based data mining.

We know that files related to salary, benefit, tax, and insurance are frequently updated together. However, if we look at the temporal file accessing characteristics, we may find that these files are accessed as high as 100% toward the end of the month (since everyone's files are updated), while they are accessed as low as 1% during other days. If we can formulate a file dependency rule that indicates salary, benefit, tax, and insurance file are usually updated toward the end of each month, for example, Friday of the 4th week of each month, we can identify the insider's malicious modification to these files occuring on days other than the Friday of the 4th week of each month. Before illustrating the enhanced model, let us first illustrate the concept of calendar schema and calendar pattern. We use the definition of calendar schema and calendar pattern presented in [2] in our model.

A ***Calendar Schema*** is denoted as $R = (d_1, d_2, \ldots, d_n)$ where each attribute d_i denotes a calendar unit. For example, the calendar schema (*month*, *week*, *day*) involves three calendar units: *month*, *week*, and *day*. A ***Calendar Pattern*** is a tuple of the form $<p_1, p_2, \ldots, p_n>$ on calendar schema R where p_i can be the wild card symbol '*'. For instance, the calendar pattern <*, 4, 5> on schema (*month, week, day*) represents Friday of the 4th week of every month.

Definition 5: **Calendar-based File Dependency**
Calendar-based File Dependency can have four forms, i.e., *Calendar-based Horizontal File Clique, Calendar-based Vertical File Clique*, *Calendar-based Horizontal File Chain*, and *Calendar-based Vertical file chain.* They are represented as (P, H), (P, V), $(P, \hat{H})$, and $(P, \hat{V})$ respectively, where P is a calendar pattern. Calendar-based file dependency specifies that a group of file are often accessed together or in certain sequence during certain period constrained by the calendar pattern P.

For instance, (<*, 4, 5>, ($s1$: salary, $\{b_1, b_2, b_3\}$: benefit, $\{t_1, t_2\}$: tax, i_1: insurance)) specifies a *calendar-based horizontal file clique*. It states that salary file: s_1, benefit files: b_1, b_2, $b3$, tax files: t_1, t_2, and insurance file: i_1 are often updated together on Friday of the 4th week of every month. If one malicious insider knows the Base Model rule that these seven files are related and tries to modify these seven files as per that rule, but, on any day except Friday of the 4th week of the month, an insider attack detection system employing this calendar-based file dependency can still detect this malicious activity and raise an alarm.

We have designed a data mining algorithm for discovering calendar-based file dependency using calendar-based association rule mining [2] and a customized calendar-based sequential pattern mining method.

The enhanced model is to be used with the base model for detecting insider threat more effectively. It is not designed to replace the base model since a significant portion of file dependencies we observed does not have calendar-based dependency characteristics.

3. CONCLUSIONS

In this paper, we illustrated a base model and an enhanced model for detecting insider threats. The base model employs a data mining approach for discovering horizontal file cliques, vertical file cliques, horizontal file chain and vertical file chains which are used to identify insider attacks. The enhanced model can be used to identify some well-crafted malicious activities based on calendar-based data mining. The prerequisite for employing the enhanced model is that the normal user activities should have a calendar-based pattern. By employing both methods proposed in the base model and the enhanced model, more insider threats can be identified.

4. ACKNOWLEDGEMENT

We are thankful to Dr. Robert. L. Herklotz for his support, which made this work possible.

5. REFERENCES

[1] G. Spanoudakis and A. Zisman, Software Traceability: A Roadmap, in Handbook of Software Engineering and Knowledge Engineering, Chang, S. K., Ed. World Scientific Publishing Co, 2005, pp. 395-428.

[2] Y. Li, P. Ning, X. Wang, and S. Jajodia, Discovering Calendar-based Temporal Association rules, In Proc. of the 8th Int'l Symposium on Temporal Representation and Reasoning, 2001.

[3] E. Cole and S. Ring. Insider Threat: Protecting the Enterprise from Sabotage, Spying, and Theft, 1 edition. Syngress, December 1, 2005.

A Traceability Link Mining Approach for Identifying Insider Threats

Yi Hu
Department of Computer Science
Northern Kentucky University
Highland Heights, KY 41099
Email: huy1@nku.edu

Brajendra Panda
Computer Science and Computer Engineering Department
University of Arkansas
Fayetteville, AR 72701
Email: bpanda@uark.edu

Overview

- In this paper, we present an insider attack detection model
  - It is designed to profile traceability links based on document dependencies and calendar-based file usage patterns for detecting insider threats
  - Document traceability links are used for detecting the insider’s malicious activity targeted at tampering the contents of files for various purposes
  - We apply the concept of traceability links in the software engineering field to this research

The base model

- Many insider threat detection system models usage patterns of individual users
  - However, the usage patterns of users often change
- Our approach concentrates on modeling file dependencies and use that for detecting insider threats
  - File dependencies rarely change
  - All files in a computer system are tagged with different categories
    - For instance, for payroll accounting systems, files can be categorized as salary, benefit, tax, insurance, etc.

The base model

- A Horizontal File Clique consists of related files that are belonged to *different* file categories and these files are often updated together
- A Vertical File Clique consists of files that are belonged to the *same* file category and these files are often updated together
- A Horizontal File Chain consists of related files that are belonged to *different* file categories and these files are often updated in certain sequence
- A Vertical File Chain consists of files that are belonged to the *same* file category and these files are often updated in certain sequence
- We designed an algorithm that employs association rule mining and sequential pattern mining for identifying these document dependencies

The base model

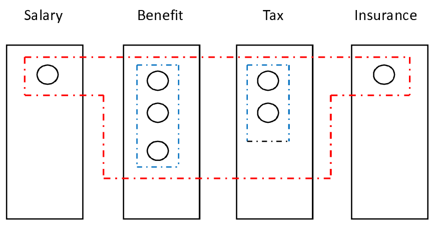

Figure 1. Vertical File Clique and Horizontal File Clique

- Two vertical file cliques and one horizontal file clique
- One vertical file clique has 3 files belonged to *Benefit* category, say, retirement plan, stock option plan, flexible spending account plan
- Another vertical file clique has 2 files of *Tax* category, state tax and federal tax, for example

The base model

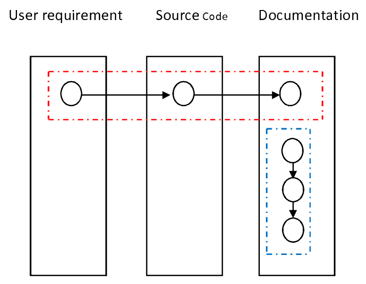

Figure 2. Vertical File Chain and Horizontal File Chain

- Figure 2 presents one vertical file chain and one horizontal file chain that are reflected in the software engineer practice
- A change in user requirement file causes a corresponding change in a source code file and the subsequent update in the documentation
  - This illustrates the relationships among three files of different categories in the horizontal file chain

The base model

- We assume the malicious insider does not have the knowledge of traceability links between the files in the base model
- If a malicious insider modifies certain files without updating other files related in the vertical or horizontal file cliques or without updating relevant files in correct sequence in the vertical or horizontal file chains, an alarm will be raised

The advanced model

- In case the malicious insider knows the document dependencies, the advanced model can help

- Calendar-based File Dependency
  - Calendar-based file dependency specifies that a group of file are often accessed together or in certain sequence during certain period constrained by the calendar pattern *P*

  - For instance, (<*, 4, 5>, (*s1*: salary, {b_1, b_2, b_3}: benefit, {t_1, t_2}: tax, i_1: insurance)) specifies a *calendar-based horizontal file clique*. It states that salary file: s_1, benefit files: b_1, b_2, *b3*, tax files: t_1, t_2, and insurance file: i_1 are often updated together on Friday of the 4th week of every month

The advanced model

- If one malicious insider knows that these seven files are related and tries to modify all of these seven files on any day except Friday of the 4th week of the month, an insider attack detection system employing this calendar-based file dependency can still detect this malicious activity and raise an alarm
- We designed a data mining algorithm for discovering calendar-based file dependency for detecting these well-crafted attacks

Conclusions

- In this paper, we illustrate a base model and an advanced model for detecting insider threats.
  - The base model employs a data mining approach for discovering horizontal file cliques, vertical file cliques, horizontal file chains, and vertical file chains which are used to identify insider attacks.
  - The advanced model can be used to identify some well-crafted malicious activities based on calendar-based data mining.