Chapter 11

DETECTING COLLUSIVE FRAUD IN ENTERPRISE RESOURCE PLANNING SYSTEMS

Asadul Islam, Malcolm Corney, George Mohay, Andrew Clark, Shane Bracher, Tobias Raub and Ulrich Flegel

Abstract As technology advances, fraud is becoming increasingly complicated and difficult to detect, especially when individuals collude. Surveys show that the median loss from collusive fraud is much greater than fraud perpetrated by individuals. Despite its prevalence and potentially devastating effects, internal auditors often fail to consider collusion in their fraud assessment and detection efforts. This paper describes a system designed to detect collusive fraud in enterprise resource planning (ERP) systems. The fraud detection system aggregates ERP, phone and email logs to detect collusive fraud enabled via phone and email communications. The performance of the system is evaluated by applying it to the detection of six fraudulent scenarios involving collusion.

Keywords: Fraud detection, collusion, enterprise resource planning systems

1. Introduction

A 2010 survey conducted by the Association of Certified Fraud Examiners (ACFE) indicated that the annual median loss per company (in the Oceania region) from fraud exceeded $600,000 [2]. A typical organization loses 5% of its annual revenue to fraud and abuse. Fraud is more complicated and increasingly difficult to detect when mid- and upper-management, who are capable of concealing fraudulent activities, collude [16]. The 2006 ACFE national fraud survey [1] notes that, while 60.3% of fraud cases involved a single perpetrator, the median loss increased from $100,000 for single-perpetrator fraud to $485,000 when multiple perpetrators colluded. It is relatively easy to identify individual fraud-

ulent transactions. However, fraud involving a combination of multiple legitimate transactions is extremely difficult to detect [6].

Enterprise resource planning (ERP) systems help prevent fraud by applying policy and internal controls. However, the effectiveness of controls is limited because they generally do not detect multi-transaction fraud. Also, controls that implement segregation of duties are often disabled in enterprises that have insufficient staff.

We have developed a system for detecting patterns of individual user fraudulent activity called "fraud scenarios" [7]. Fraud scenarios are a set of user activities that indicate the possible occurrence of fraud. Fraud scenarios parallel computer intrusion scenarios, and the fraud detection system operates in a similar manner to a signature-based intrusion detection system. However, fraud scenarios differ from intrusion scenarios in that they focus on high-level user transactions on financial data rather than computer system states and events. There is a correspondingly greater degree of system independence in fraud detection, which can be exploited by separating the abstract or semantic aspects of fraud signatures from their configuration aspects [7]. For this reason, we have designed a language specifically for defining fraud scenarios.

The signature language semantics have been tested using fraud scenarios in ERP systems. The scenarios reflect segregation of duty violations and instances of masquerade. Segregation of duty violations are detected by identifying multiple transactions conducted by a single individual. Masquerade scenarios are detected by identifying multiple transactions carried out by supposedly different individuals from the same terminal.

The fraud detection system performs *post facto* (non-real-time) analysis and investigation. The detection of a scenario does not confirm that fraud has occurred; rather, it identifies a possible occurrence of fraud that requires further investigation.

The extended fraud detection system described in this paper aggregates ERP, phone and email logs, and analyzes them to identify various forms of potentially collusive communications between individuals. Six scenarios that express possible collusion are considered: Redirected Payment (S01); False Invoice Payment (S02); Misappropriation (S03); Non-Purchase Payment (S04); Anonymous Vendor Payment (S05); and Anonymous Customer Payment (S06).

2. Related Work

Automated fraud detection significantly reduces the laborious manual aspects of screening and checking processes and transactions in order to control fraud [12]. Businesses are highly susceptible to internal fraud

perpetrated by their employees. Internal fraud detection concentrates on detecting false financial reporting by management personnel [4, 5, 10, 15] and anomalous transactions by employees [8, 9].

Data mining and statistical approaches are often used to detect suspicious user behavior and anomalies in user activity logs and transaction logs [12]. An alternative approach to anomaly detection is a process that creates fraud scenarios and identifies a means to detect each scenario [7]. Although such signature matching approaches are not widely used in fraud detection, they are commonly used in intrusion detection systems [11, 13, 14]. The fraud detection system described in this paper integrates and analyzes accounting transaction logs and user activity logs, and uses signature matching to detect collusive fraud.

3. Definition and Detection of Fraud Scenarios

A fraud scenario includes a scenario name, description, list of components, attributes and scenario rules. A component is a transaction (extracted from a transaction log) or another previously-defined scenario. Scenario attributes hold the values that define the behavior and characteristics of the scenario, in particular, the "inter-transaction" conditions that capture the essential nature of the fraud.

Scenario rules describe the order and timing of the occurrence of each component as it pertains to the fraud. The rules contain the minimum or maximum time intervals allowed between component occurrences. Each time constraint corresponds to one of three levels: default, scenario or component level. Default values are used by the fraud detection system when values are not specified in a scenario definition. Scenario level values apply over all components while component level values apply between two specific components in a scenario. Component level values override default and scenario level values.

3.1 Defining Fraud Scenarios

A scenario definition file (SDF) is an XML file that specifies and stores scenario definitions. For example, the Redirected Payment (S01) scenario shown in Figure 1 describes the behavior of making a payment in such a way that the payment goes to a redirected account instead of the vendor's actual account. The scenario involves making payments to a vendor account after changing the bank account details of the vendor to a different account. After the payment is made, the user changes the bank details back to the original values.

In the case of the Change_Vendor_Bank scenario, three transaction codes (FK02, FI01 and FI02) may appear in the transaction log. The

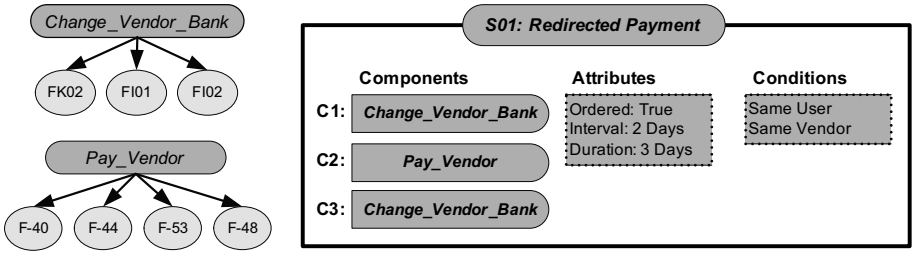

Figure 1. Redirected Payment (S01).

scenario matches any of these three transaction codes. Similarly, the Pay_Vendor scenario matches any of four transaction codes (F-40, F-44, F-48 and F-53) in the transaction log. Change_Vendor_Bank and Pay_Vendor correspond to components. The components match individual transactions or events (not groups or sequences of transactions or events) in the source logs. The signature scenarios for detecting fraudulent activities consist of sequences of multiple transactions/components.

This paper uses four fraud scenarios [7] to describe the process of extending individual fraud scenarios to include collusion. The scenarios are: Redirected Payment (S01), False Invoice Payment (S02), Misappropriation (S03) and Non-Purchase Payment (S04). These scenarios are modified to include additional requirements for detecting communications between colluding individuals.

3.2 Defining Collusive Fraud Scenarios

The fraud detection system uses three source logs: ERP system logs, phone logs and email logs. The ERP system logs contain information about the day-to-day activities of users. The phone and email logs maintain information about the communications between users. Note that to preserve privacy, the content of the communications is not analyzed, only that direct communications have occurred between the parties of interest. Other potential sources of information are door logs, office layouts (to identify people working in the same location) and personal relationships.

Figure 2 presents an extension of S01 that includes collusion. The extension involves collaborators contacting each other between steps by phone or email (Phone_or_Email).

The False Invoice Payment with Collusion (S02_col) scenario involves the creation, approval and payment of a false invoice (Figure 3). The presence of any of the three transaction codes FB60, MIRO or F-43 in the transaction log indicates the creation of an invoice (Create_Invoice). The

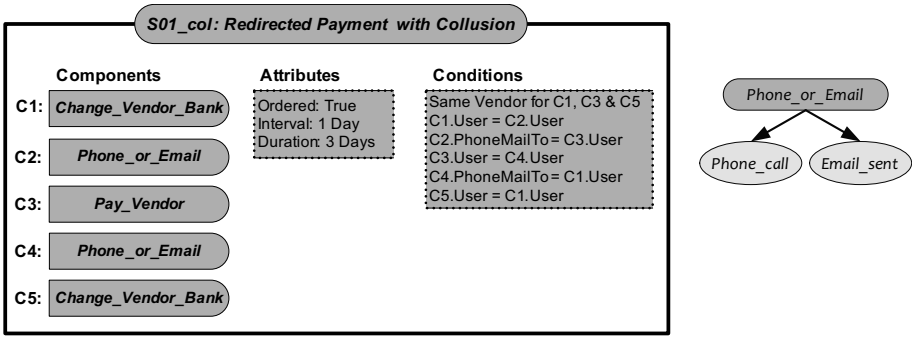

Figure 2. Redirect Payment with Collusion (S01_col).

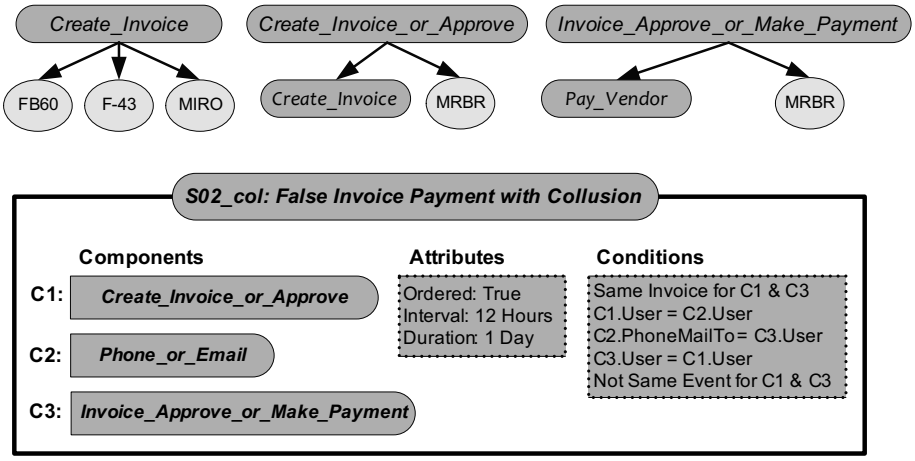

Figure 3. False Invoice Payment with Collusion (S02_col).

second activity, which involves the approval of an invoice, is indicated by the MRBR transaction code. The third activity, making a payment, uses the already-defined component Pay_Vendor from scenario S01. Figure 3 shows the sequence of components that occurs for the same purchase order with a Phone_or_Email component between them, which indicates communications between colluding users.

The Misappropriation with Collusion (S03_col) scenario involves the misappropriation of company funds. The fraud involves the creation of a purchase order and the approval of the purchase. In particular, fraud may exist when a party who has the authority to make a purchase order colludes with another party who has the authority to approve it. Scenario S03_col is the sequence of two components, Create_PO and PO_Approval, for the same purchase order with a Phone_or_Email com-

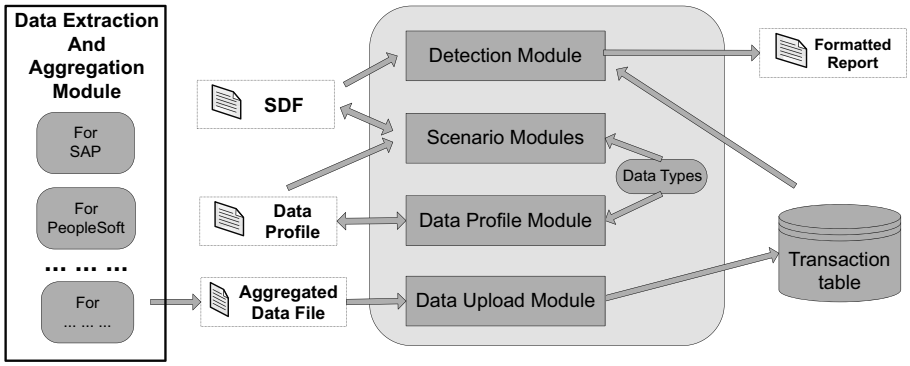

Figure 4. Detection process.

ponent between them that indicates collusion between the two parties. The necessary conditions for collusion are similar to those shown in Figure 3.

The Non-Purchase Payment with Collusion (S04_col) scenario involves the generation of a purchase record in the system and a payment being made without the purchase actually occurring. The fraud may potentially exist when one party creates a purchase order or a goods received transaction, another party creates an invoice on the same purchase order, and communications exist between the two parties. Scenario S04_col is the sequence of the Create_PO_or_Good_Receipt and Create_Invoice components for the same purchase order with a Phone_or_Email component between them that indicates collusion.

4. Detecting Collusive Fraud Scenarios

Figure 4 presents the process for detecting fraud scenarios. The process uses the fraud scenario definitions in the SDF and searches for matches in the aggregated log data. The search process generates an SQL query based on the scenario definition and runs the query against the aggregated data file. Query matches are flagged as possible fraud events.

The first component is the data extraction and aggregation module, which extracts and aggregates log data from different sources. The module is located external to the main detection process to allow for extensibility and accommodate ERP system changes and additional system logs. The current implementation extracts transactions from an SAP R/3 ERP system.

The data profile contains the description of the data file, number of fields, field types, user-defined names for fields, column and line separa-

tors, fields to be considered and fields to be ignored. The format and types of the individual fields in the aggregated data file are defined by the user. Several pre-defined data types are provided (e.g., date, time and event), and users can add or modify the list as needed. In the data profile, users can optionally define the information extraction process for each field from a specific position (e.g., a VendorID is a four-digit string in the second position of the fourth field).

The data upload module uploads data from the aggregated data file to a database. The data profile must be defined by the user at the time of upload. The data upload module creates a transaction table according to the data profile and uploads data to the table.

The SDF specifies the known fraud scenarios that are used in searches of the aggregated data file. Users may add new scenarios or edit existing scenarios by changing the data profile and list of data types. Interested readers are referred to [7] for details about fraud scenario creation and specification.

5. Experimental Validation

The fraud detection system detects fraudulent activities by matching the scenarios against transactions or events recorded in the database. This section evaluates the ability of the implemented system to detect collusive fraud. The collusive fraud scenarios described involve phone and email communications. Thus, the data extraction module extracts phone and email logs as well as user transaction logs from an ERP system.

An SAP R/3 ERP version 4.0 system served as the source of transaction log data. Fraud detection requires at least three data columns in each transaction record: timestamp, user ID and event. Depending on the scenario, additional fields may be of interest. For example, detecting the Redirected Payment (S01) scenario in the Change_Vendor_Bank component requires matching the vendor identification number between components.

A major problem in fraud detection research is the lack of real-world data for testing. Additionally, real-world background data is unavailable, which makes it difficult to integrate synthetic fraudulent data with legitimate background data [3]. Therefore, our testing used synthetic transaction data that was generated randomly.

The test data comprised 100,000 records corresponding to 100 users and 100 terminals. It was assumed that a user does not always use a specific terminal in order to model masquerade scenarios where users perform activities from multiple terminals or where multiple users perform

Table 1. Activities in randomly-generated data.

Scenario	Matches
Change_Vendor_Bank	2,730
PO_Approval	5,140
Pay_Vendor	10,820
Good_Receipt	15,570
Create_Invoice	5,280
Create_Vendor	21,510
Invoice Approval (MRBR)	2,430
Create_Customer	4,830
Create_PO	13,190
Credit_to_Customer	5,260
Phone_or_Mail	15,970

activities from one terminal. The synthetic data incorporated vendor identification numbers, invoice numbers, purchase order numbers and customer identification numbers. Email and phone call log data were also generated randomly.

Each instance of fraud identified by the system was analyzed to verify its correctness. As a secondary check for each instance, records were added and deleted and the scenario detection process re-executed. This provided a means to verify that adding relevant transactions had the expected effect of producing a match where none existed previously, and that deleting records had the effect of producing no match where one previously existed.

Table 1 lists the numbers of activities or components present in the randomly-generated data. Note that the summation is greater than 100,000 because some transaction codes are present in multiple activities. For example, the Create_Vendor activity includes all the transaction codes that indicate the creation of a new vendor and any editing of a vendor record; thus, the transaction codes for the Change_Vendor_Bank activity are also included in the Create_Vendor activity.

The Change_Vendor_Bank component matches 2,730 records and the Pay_Vendor component matches 10,820 records. Testing of the Redirected Payment (S01) scenario (without user collusion) involves locating sequences of Change _Vendor_Bank, Pay_Vendor, Change_Vendor_Bank for the same vendor when the maximum interval between any two activities is two days and the overall scenario duration is less than three days. The search took 1.2 seconds on a Pentium 4 machine with 2 GB RAM running Microsoft Windows XP Professional. Two matches were

Table 2. Results for the S01 scenario.

Time	Trans.	User	Term.	Vendor	Invoice	P.O.
10:17:53	FK02	U010	T04	V00020		
11:32:17	F-53	U010	T05	V00020	I00024	P00000010
13:02:48	FK02	U010	T10	V00020		
01:03:34	FK02	U009	T02	V00001		
02:28:50	F-53	U009	T06	V00001	I00017	P00000004
02:58:40	FK02	U009	T03	V00001		

identified (Table 2). One record was returned for each match; however, for clarity each match is displayed across three rows.

Table 3. Results for the S01_col scenario.

Time	Trans.	User	Term.	Recip.	Vendor	Invoice	P.O.
09:57:55	FK02	U007	T04		V00006		
10:59:44	PhoneTo	U007	T09	U002			
11:39:39	F-48	U002	T06		V00006	I00039	P00000013
12:15:07	MailTo	U002	T10	U007			
13:00:22	FK02	U007	T03		V00006		

Testing of the Redirected Payment with Collusion (S01_col) scenario yielded four matches in 92 seconds, one of which is shown in Table 3. The results demonstrate that considering collusion reveals additional instances of potential fraud.

Table 4. Results for all six scenarios.

	Without Collusion		With Collusion	
Scenario	**Matches**	**Time**	**Matches**	**Time**
Redirected Payment (S01)	2	1.2s	4	92s
False Invoice Payment (S02)	15	1.1s	5	18s
Misappropriation (S03)	5	0.8s	6	17s
Non-Purchase Payment (S04)	27	2.3s	9	5.6s
Anonymous Vendor Payment (S05)	42	1.3s	21	3.6s
Anonymous Customer Payment (S06)	0	0.3s	2	2.7s

Table 4 presents the results corresponding to all six scenarios. The time taken by a query for a collusive scenario is greater than that for its non-collusive counterpart. There are two reasons. The first reason is that collusive scenarios involve more data than non-collusive scenarios. Second, queries for collusive scenarios involve cross-field database joins

rather than same-field joins in the case of non-collusive scenarios. For example, in a collusive scenario, the query matches conditions between fields in which one individual's user name is matched to another via an email or phone call, and both are matched to vendor or purchase order activity.

6. Conclusions

Fraudulent activity involving collusion is a significant problem, but one that is often overlooked in fraud detection research. The fraud detection system described in this paper analyzes ERP transaction logs, and email and phone logs of user communications to detect collusion. The system permits the configuration of scenarios, supporting focused analyses that yield detailed results with fewer false positive errors.

Our future research will test the fraud detection system on real SAP system data. In addition, the detection methodology will be extended to enable the system to operate in real time.

Acknowledgements

This research was supported by the Australian Research Council and by SAP Research.

References

[1] Association of Certified Fraud Examiners, 2006 ACFE Report to the Nation on Occupational Fraud and Abuse, Austin, Texas (www.acfe.com/documents/2006RTTN.ppt), 2006.

[2] Association of Certified Fraud Examiners, 2010 Report to the Nation on Occupational Fraud and Abuse, Austin, Texas (www.acfe.com/rttn/2010-rttn.asp), 2010.

[3] E. Barse, H. Kvarnstrom and E. Jonsson, Synthesizing test data for fraud detection systems, *Proceedings of the Nineteenth Annual Computer Security Applications Conference*, pp. 384–394, 2003.

[4] T. Bell and J. Carcello, A decision aid for assessing the likelihood of fraudulent financial reporting, *Auditing: A Journal of Practice and Theory*, vol. 19(1), pp. 169–184, 2000.

[5] K. Fanning and K. Cogger, Neural network detection of management fraud using published financial data, *Intelligent Systems in Accounting, Finance and Management*, vol. 7(1), pp. 21–41, 1998.

[6] D. Coderre, *Computer Aided Fraud Prevention and Detection: A Step by Step Guide*, John Wiley, Hoboken, New Jersey, 2009.

[7] A. Islam, M. Corney, G. Mohay, A. Clark, S. Bracher, T. Raub and U. Flegel, Fraud detection in ERP systems using scenario matching, *Proceedings of the Twenty-Fifth IFIP International Conference on Information Security*, pp. 112–123, 2010.

[8] R. Khan, M. Corney, A. Clark and G. Mohay, A role mining inspired approach to representing user behavior in ERP systems, *Proceedings of the Tenth Asia Pacific Industrial Engineering and Management Systems Conference*, pp. 2541–2552, 2009.

[9] J. Kim, A. Ong and R. Overill, Design of an artificial immune system as a novel anomaly detector for combating financial fraud in the retail sector, *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 405–412, 2003.

[10] J. Lin, M. Hwang and J. Becker, A fuzzy neural network for assessing the risk of fraudulent financial reporting, *Managerial Auditing Journal*, vol. 18(8), pp. 657–665, 2003.

[11] C. Michel and L. Me, ADeLe: An attack description language for knowledge-based intrusion detection, *Proceedings of the Sixteenth IFIP International Conference on Information Security*, pp. 353–368, 2001.

[12] C. Phua, V. Lee, K. Smith and R. Gayler, A comprehensive survey of data mining based fraud detection research (arxiv.org/abs/1009.6119v1), 2010.

[13] P. Porras and R. Kemmerer, Penetration state transition analysis: A rule-based intrusion detection approach, *Proceedings of the Eighth Annual Computer Security Applications Conference*, pp. 220–229, 1992.

[14] J.-P. Pouzol and M. Ducasse, From declarative signatures to misuse IDS, *Proceedings of the Fourth International Symposium on Recent Advances in Intrusion Detection*, pp. 1–21, 2001.

[15] S. Summers and J. Sweeney, Fraudulently misstated financial statements and insider trading: An empirical analysis, *The Accounting Review*, vol. 73(1), pp. 131–146, 1998.

[16] J. Wells, *Corporate Fraud Handbook: Prevention and Detection*, John Wiley, Hoboken, New Jersey, 2007.