Fraud Detection in Mobile Payments Utilizing Process Behavior Analysis

Roland Rieke, Maria Zhdanova, Jürgen Repp, Romain Giot, Chrystel Gaber

Fraud Detection in Mobile Payments Utilizing Process Behavior Analysis

Roland Rieke*†, Maria Zhdanova†, Jürgen Repp†, Romain Giot‡ and Chrystel Gaber‡
*Philipps-Universität Marburg, Germany
†Fraunhofer SIT, Darmstadt, Germany
Email:{roland.rieke,maria.zhdanova,juergen.repp}@sit.fraunhofer.de
‡France Télécom-Orange Labs,
Caen, France
Email:{romain.giot,chrystel.gaber}@orange.com

***Abstract*—Generally, fraud risk implies any intentional deception made for financial gain. In this paper, we consider this risk in the field of services which support transactions with electronic money. Specifically, we apply a tool for predictive security analysis at runtime which observes process behavior with respect to transactions within a money transfer service and tries to match it with expected behavior given by a process model. We analyze deviations from the given behavior specification for anomalies that indicate a possible misuse of the service related to money laundering activities. We evaluate the applicability of the proposed approach and provide measurements on computational and recognition performance of the tool – Predictive Security Analyzer – produced using real operational and simulated logs. The goal of the experiments is to detect misuse patterns reflecting a given money laundering scheme in synthetic process behavior based on properties captured from real world transaction events.**

***Keywords*—money laundering; predictive security analysis; analysis of business process behavior; security modeling and simulation; security monitoring.**

I. Introduction

The field of Mobile Money Transfer (MMT) is a growing market segment, particularly in developing countries where banking systems may not be as dense or available as in developed countries. For example, M-Pesa, which was launched in 2007 in Kenya, displayed in December 2011 about 19 million subscribers, namely 70% of all mobile subscribers in Kenya [1]. Orange Money is deployed in 10 countries and gathers around 14% of the mobile subscribers of these countries [2]. In such services, transactions are made with electronic money, called mMoney. The users can convert cash to mMoney through distributors and use it to purchase goods at merchants, pay bills or transfer it to other users [3]. Like any other money transfer service, this service is exposed to the risk of money laundering, i.e., misuse through disguising illegally obtained funds to make them seem legal, and more generally fraud risk that implies any intentional deception made for financial gain [3].

In this paper, we apply a tool – the Predictive Security Analyzer (PSA) – that implements predictive security analysis at runtime [4], [5] in order to identify misuse patterns in event streams of MMT transactions that can be concerned with money laundering activity. Predictive security analysis at runtime is an advanced method for the evaluation of security-related events and their interpretation with respect to: (1) the known control-flow of the processes involved, and (2) the required security properties. With respect to (1), deviations of observed process behavior from the given process specification are analyzed. These deviations can be the result of an evolution in the process specification, problems with the measurement (e.g., lost events), or anomalies caused by attacker's interventions. Regarding (2), continuous monitoring of security properties specified for the process in question is performed to detect potential security violations.

The PSA takes as an input real-time events from the process execution environment, a process model and its security requirements. For informal modeling of MMT processes (at the business logic level) we employ the Event-driven Process Chain (EPC) language [6]. Security analysis of event-driven processes uses a formal process model encompassing the incoming events. If a critical state, i.e., a process anomaly or a security requirement violation, is detected the PSA provides a semi-automatic treatment by visualization and inspection of the problem (uncertainty management) or performs automatic generation and dissemination of alerts for further processing depending on the selected option. Due to the advanced security analysis method the PSA can enhance evaluation and correlation capabilities of Security Information and Event Management (SIEM) systems, as shown for MASSIF SIEM [7].

In this paper we evaluate the applicability of the approach and the performance of the PSA in this context. In particular, we show that: (1) the PSA is able to manipulate an event stream of operational systems in real time; (2) the PSA is able to raise alerts on most fraudulent transactions related to money laundering. The results of the experiments on real and simulated events for several money laundering scenarios will allow us to determine sensitivity and specificity of the PSA and refine the detection scheme.

The paper is organized as follows: Section II introduces the MMT scenario and a misuse case related to money laundering and Section III gives an overview of the PSA and its application in the experiment. Section IV introduces the experimental setup, while Section V discusses the results of the experiments. Section VI reviews related work and Section VII presents conclusions and directions for further research.

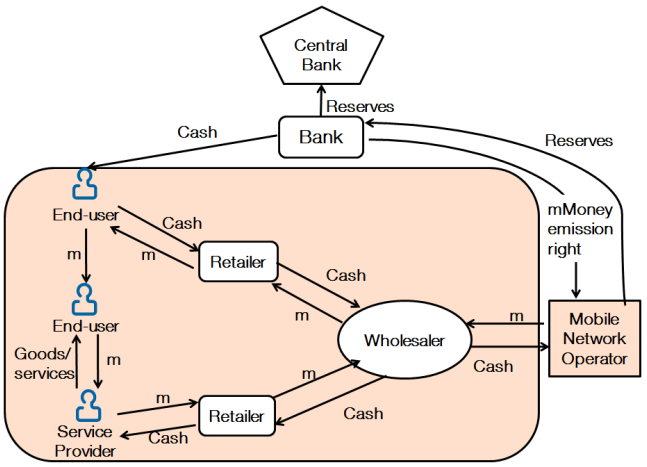


Figure 1. Economical environment of Mobile Money Transfer services

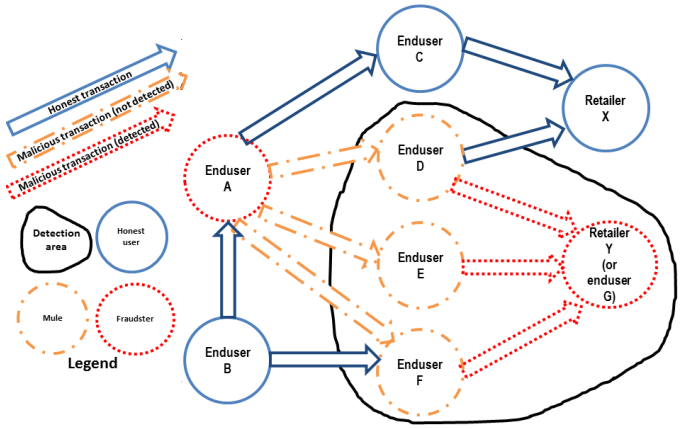


Figure 2. Description of the money laundering scenario: several mules receive mMoney from a malicious user and transfer a high percentage of this mMoney to another malicious user. The first malicious user does not want to be directly linked with the second malicious user

II. Fraud Detection in Mobile Money Transfer System

A. Mobile Money Transfer

This article is based on the MMT use case detailed in [8]. This section sums up the major points to understand the use case. MMT systems are systems where electronic money, called mMoney (or m), is issued to different roles (e.g., regular users, retailer, merchant, etc), in order to perform various types of transactions (e.g., mobile recharge, cash in, cash out, national or international transfer, bill payments, etc). Figure 1, which is adapted from [9], shows the economic principle of mMoney and the roles of the various actors. As depicted, the Mobile Network Operator (MNO) emits mMoney in partnership with a private bank. The MNO regularly produces compliancy reports to the Central Bank, responsible for the country's monetary policy. In particular, it is the MNO's responsibility to detect suspicious money laundering activities and to report them to the Central Bank, hence the importance of SIEM systems for MMT systems. The emitted mMoney can only be used among the MNO's clients who subscribe to the MMT service. The subscribers are end-users, service providers or retailers. They hold a prepaid account stored on a platform and accessible via the MNO's network and an application on their mobile device. Some users, such as retailers or service providers, can use computers to access their account. This account contains mMoney which can be acquired from the retailers. End-users can either transfer money to other end-users or purchase goods.

B. Misuse case

We are interested in a misuse case related to money laundering using MMT (see Figure 2). A malicious user (fraudster) transfers small amounts of money to several mules. These mules can optionally receive the money at the end of a chain of mules. Then the mules of the final chain make a final transfer to a second malicious user. Each mule may keep a small percentage of the transfer as a salary. Also, these mule transfers can be manually operated by a mule or automatically transfered by a malicious software installed on the mobile phone and exploiting a flaw in the mobile money application. This way, the first malicious user has transfered money to the second malicious user, but there is no direct transfer traces between them.

Many schemes of money laundering are known and criminals struggle to invent new ones [10], [11]. In this paper we consider a money laundering scheme involving the following assumptions: (i) there is only one mule in the chain of mules; (ii) the amount of a fraudulent transaction is much smaller then the average on the system; (iii) along with fraudulent actions the mules perform regular mMoney transfers; (iv) normal behavior of a MMT user (as observed in the operational logs) is persistent in regard to transaction amounts used, i.e. sudden changes in transferred mMoney amounts indicate an anomaly. These limitations do not restrict the proposed approach to fraud detection as far as one is able to model a process workflow related to a chosen (any other) money laundering scheme.

III. Predictive Security Analysis at Runtime

The PSA provides support for the whole cycle of event-driven process security analysis.

a) Event pre-processing: Events come from an Extensible Markup Language (XML) stream or a database based on a pre-defined event schema, which is necessary in order to filter out data containing information not relevant to security analysis. Therefore, the PSA supports the creation of an event abstraction and mapping of events to the corresponding process instance (cf. Figure 3).

b) Process specification, adaptation, and close-future behavior analysis: The PSA uses Asynchronous Product Automata (APA) for formal process representation [12]. Process specifications given in EPC can be modeled by APA. EPCs are used in business process engineering, deployment and runtime control and supported by such leading products as SAP R/3 and ARIS [13]. The EPC language is intended to be easy to understand and use by people who are not familiar with formal specification methods. Therefore, the PSA assists the user not only in generation of a EPC, but also in transformation of this informal process model into an operational formal model. Existing EPC models can be adapted step-by-step, using archived event logs for replay and also interactively

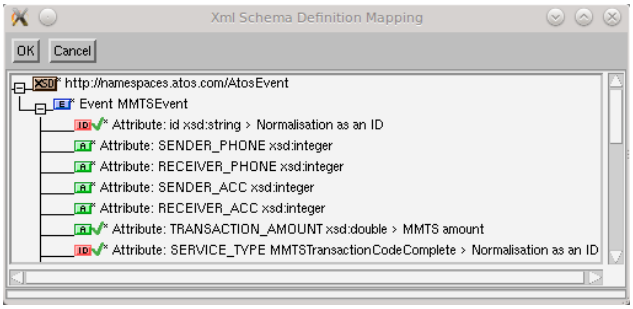


Figure 3. Feature selection. Few fields of the events are used, and the concatenation of two of them provides the process id

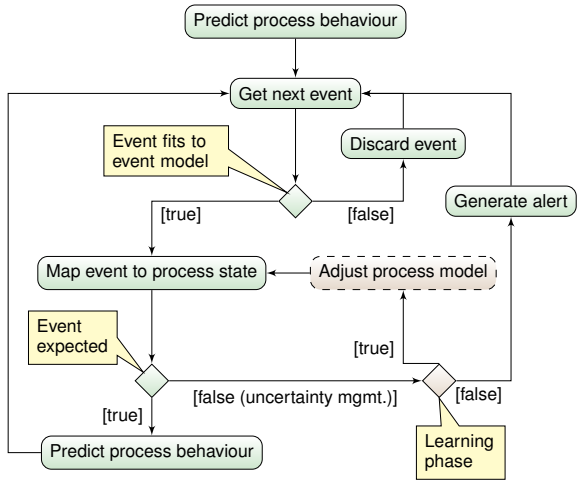


Figure 4. Runtime behavior of the PSA

during runtime. The *uncertainty management* supports semi-automatic adaptation of process models according to the context conditions. Uncertainty situations can occur during synchronization of the state of a running process instance and the state of the model if the process model is not accurate enough or outdated, or when unknown events are received or expected events are missing. Beside APA models, Petri Net Markup Language (PNML) specifications [14] generated by process discovery tools (e.g., ProM [15]) can be imported. For all specification methods the computation of the close-future behavior is supported.

c) Security requirements specification and evaluation: The PSA allows for specification of the required security properties that the monitored process should fulfill (a security model in form of monitor automata), on-the-fly check of security requirements with respect to current process behavior (detection of critical process states), as well as techniques for on-the-fly check of security requirements with respect to close-future process behavior (prediction of critical process states).

d) Situational awareness and alarm generation: The PSA provides visualization of current process states with respect to security requirements by means of security monitors, and generation of alarms on detection of critical states.

In the work presented in this paper the PSA has been used in two phases, namely, a *learning phase* and an *anomaly detection phase*.

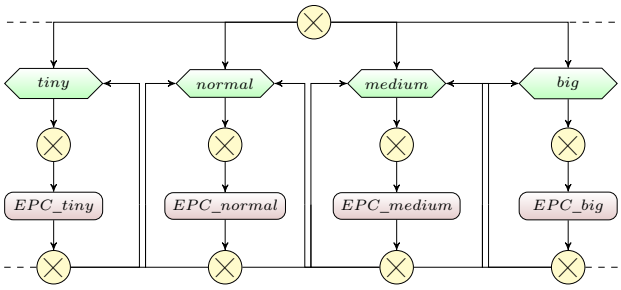


Figure 5. Subgraph of EPC for MMTS

In the initial learning phase, the normal behavior pattern with regard to the transaction characteristics is learned by processing an event log without malicious content. A mapping to classify the transactions with regard to the amount of money transferred (cf. Table I) and the order of such abstract events in the event log is used for this purpose. For example, an event from the MMT system where the amount of transferred money is greater than 500 but less or equal 1000 is mapped to the abstract event $medium$. This mapping is created empirically using real operational logs of the MMT system and can change if different training sets are used. Figure 4 shows an overview of the PSA event processing steps in the learning phase. The dashed action "Adjust process model" is done semi-automatically and requires the user's involvement during runtime. All other actions are performed automatically.

Figure 5 shows a subgraph of an EPC which was learned for the MMT model. The graph shows the control flow structure of a process as a chain of events and functions. Rectangles with rounded corners denote EPC functions and hexagons denote EPC events. Functions represent active components, i.e., activities, tasks or process steps, which are triggered by events. Events are passive, they represent the occurrence of a state which describes the situation before, or after, a function is executed. Logical and, or, and xor (exclusive or) operators are used to connect the basic constructs, in this way the control flow is specified. For example, after an event $medium$ the function EPC_medium is triggered. The expected "normal" events after execution of EPC_medium are $\{normal, medium, big\}$.

In the anomaly detection phase the PSA identifies deviations from the normal characteristics based on the values from a transaction monitor and generates alerts. In our experimental setup, the transaction monitor has been replaced by synthetic process behavior composed of simulated logs based on properties captured from real logs. In the anomaly detection phase the dashed action "Adjust process model" in Figure 4 is not used. Instead, an alert is generated automatically by the uncertainty management components of the PSA.

In order to reduce the number of alarms, it is necessary to configure the normal behavior model with the help of a real world dataset with annotated transactions (suspicious/not suspicious) or to use an additional component which filters the alerts generated by the PSA. Fraud analysis with additional components is beyond the scope of this paper.

IV. Experimental Protocol

A. PSA Configuration

The PSA is used in a non-interactive way: when it detects an unexpected event, an alert is automatically generated.

1) Definition of a mapping: The amount of a transaction is a continuous variable in $\mathbb{R}$, therefore discretization is necessary to get computable abstractions of the behavior. For this reason, we have empirically created various classes of transfer amounts (see Table I).

2) Definition of an EPC: In order to use the PSA it is necessary to define an EPC which models the MMT process. However, the events generation does not come from the control flow of the mobile money system, but from the behavior of its users. For this reason, the EPC must model the workflow of the user and not the workflow of the system. It is possible to define several processes in one PSA model. The PSA also provides the capability to investigate parallel running process instances with the same behavior. This capability was used to specify a general behavior which is followed by every user.

We found out that it is challenging to define a workflow of transactions because every user is free to use the system as he wants (*i.e.*, he can choose own amounts, frequencies, communities of interests, etc.). There is one process instance for each pair of active users and type of transaction (*i.e.*, $(user_1, CASHIN)$), because we make the assumption that the amount of transactions of the same type (*i.e.*, only $CASHIN$, only $TRANSFER$, etc.) are similar, while amounts of transactions of different kinds are not [16].

For this reason we have to create a more general process. A process behavior representation which is generated by the PSA from the respective EPC (cf. Figure 5) is shown in Figure 6. Each node is a state, each edge is a transition labeled with the event source. For each state, only the following transactions are authorized: do a transfer in the same amount family, do a transfer with the previous amount family, or do a transfer with the next amount family. The very first transition allows to go to any state. All the other possible transactions which are not present in the graph raise an alarm and are considered as being potentially malicious.

B. Logs at our disposal

1) Operational logs: Although different kinds of logs (access, transfer, etc.) can exist in the mobile money transfer system, we only have at our disposal the transactions log. The transactions log contains the sender and receiver of the transaction, its amount, its success, the type of transaction and the type of the sender and of the receiver as well as other fields specifics to the system. These logs are driven by the behavior of the users: indeed the events are propagated only when the user do some transactions. We have more than 4.5 millions of correct events (accepted transactions) acquired on a period of 9 months.

However, as we have no ground truth (*i.e.*, fraudulent or not fraudulent) on these events, we cannot use them directly to detect fraud. They are nevertheless very useful to analyze the ability of the PSA to manage real life events in real time. For detection evaluation (in terms of error rates), we use simulated logs.

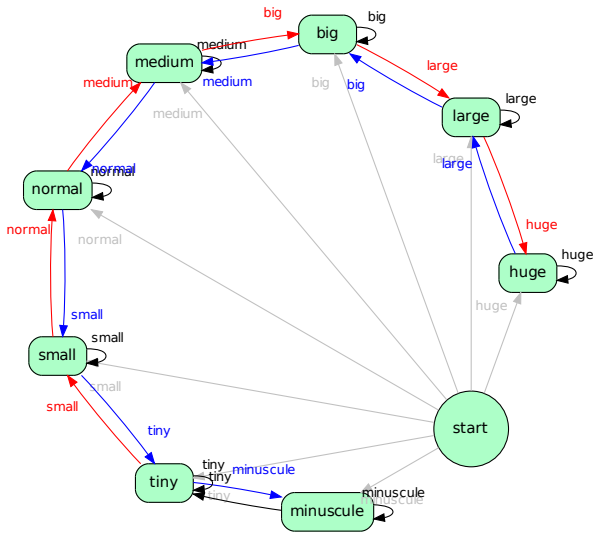


Figure 6. Process behavior representation of the EPC defined for MMTS

2) Simulated Events: As real world transactions have no ground truth we have implemented the misuse case in a simulator [17]. This simulated world is based on properties captured from real world transaction events. We are interested in a money laundering scenario (see Figure 2).

We have configured 3 scenarios with different number of users of the following categories:

a) Regular users: They make regular transfers (mean amount of 4000, standard deviation of 500), withdrawal and deposits.

b) Malicious users: (within the group of regular users). They want to exchange money without leaving direct traces in the system.

c) Mules: (within the group of regular users). They receive an amount (min amount of 20, max amount of 100) from a malicious user and transfer it later to another one after keeping 10% of interest.

d) Merchants: They correspond to shops where regular users can buy goods with mMoney.

e) Retailers: They allow end user to exchange mMoney with real money (and vice versa).

The parameters of the three scenarios are:

S1 No money laundering: 50 regular users, 8 merchants, 4 retailers. This set serves to verify if normal transfers are detected as being fraudulent (False Positive).

S2 Money laundering: 50 regular users, 5 mules, 8 merchants, 4 retailers. This set serves to verify if the PSA is able to detect frauds.

S3 Money laundering with more individuals: 500 regular users, 10 mules, 16 merchants, 4 retailers This set serves to verify if the PSA can detect frauds when there are more non-fraudulent events.

Table I. MAPPING FOR DISCRETIZATION OF THE TRANSACTIONS' AMOUNTS

Class	minuscule	tiny	small	normal	medium	big	large	huge
Amount condition	≤ 5	≤ 50	≤ 200	≤ 500	≤ 1000	≤ 2000	≤ 5000	other

Note that, we have used a slightly different mapping for the simulated logs in comparison to the real logs.

C. Experiment

Our analysis tackles the computational and recognition performance of the PSA. We seek to ascertain how long it takes to process an event or spread an alert and whether it is possible to treat in real time the stream of events of an operational MMT system. The computations were done with a personal computer (2 cores CPU at 2.70GHz, 4Gb RAM). Finally, we will measure the error rates of the PSA. Any potential errors in detection will be taken advantage of in order to learn and improve the detection scheme.

D. Evaluation metrics

The computational performance of the PSA is evaluated by counting the number of events per second successfully processed by the PSA. For the recognition performance we use several metrics: (a) *False Positive*, not fraudulent event is detected as being fraudulent; (b) *False Negative*, fraudulent event is detected as being not fraudulent; (c) *True Positive*, fraudulent event is detected as being fraudulent; (d) *True Negative*, not fraudulent event is detected as being not fraudulent.

V. EXPERIMENTAL RESULTS

A. Real events analysis

For the real events, we are interested in the computational performances. Figure 8a represents the number of transactions between the different states (only the transactions present in the events log are displayed). Most transactions are not considered as suspicious and most suspicious transactions are between the state "tiny" and "normal" (so a more accurate process model would allow such kind of transitions).

We found out that there was one process per pair of users and transaction type. With the real log, the PSA was able to manage 640,000 instances without any problem. 40 minutes were enough to process 4.5 millions of events, with the process behavior presented in Figure 8a, and produce 0.5 millions of alerts. 33 minutes were enough for a complete run which does not generate alerts. Set X as the time to process an event and Y the additional time to process an alert. They can be found by solving these two equations: $X * 4.5M = 33 * 60$ and $X * 4.5M + Y * 0.5M = 40 * 60$, which gives $X = 0.00044$ and $Y = 0.00480$. Thus, in the best theoretical case (no alerts are generated), the PSA is able to process more than 2200 events/second $(1/X)$, while in the worst theoretical case (all events raise an alert), the number is reduced to 191 events/second $(1/(X+Y))$. We can summarize these results by saying that the PSA is able to manage an average of 100 millions of events per day on a standard computer $((2200 + 191)/2 * 60 * 60 * 24)$.

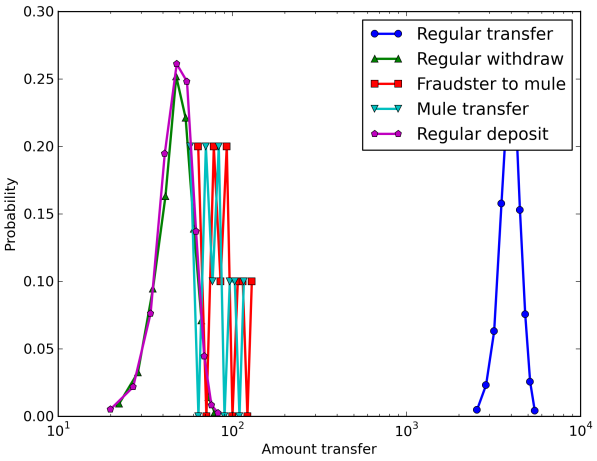


Figure 7. Histogram of the transactions amount of scenario S3

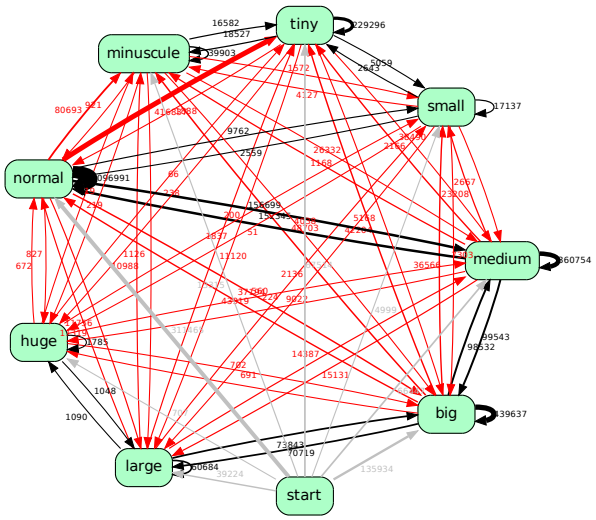
B. Simulated events analysis

In case of the simulated events, we are interested in the recognition performances. As the simulation is stochastic, the evaluation has been repeated several times. However, the results are very similar for each run.

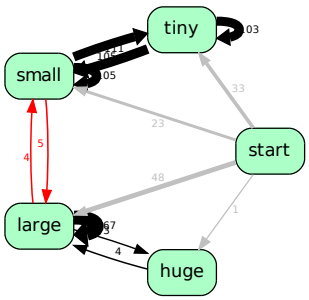
Figure 7 presents the histogram of the transactions amounts depending on their type in scenario S3. We can see that the distribution of the mule transfer is a bit translated to lower amounts in comparison to the distribution of initial fraudster's transfers (because of the interest kept by the mule) and overlaps with the distribution of withdrawal and deposits.

Figure 8b represents the number of transactions between the different states in scenario S2. For the simulated logs, we can see that 9 (4+5) transactions have been detected as suspicious. However, most of the transactions are in the range of authorized transactions according to the EPC.

Table II gives the recognition performance of the PSA on the selected example of each scenario, and Figure 9 presents the transactions of the participants involved in the fraud of S2 (the other users which made no transactions with the fraudsters are not displayed for clarity reasons). Each node represents a user, each edge represents a transaction, the label of an edge represents the index of the transaction in the sequence. Small gray edges represent True Negative, orange (gray for printed version) edges represent False Positive, green (dark gray) edges represent True Positive, red (black) edges represent False Negative. For scenario S1, the ratio of False Positive is null. For scenario S2, the ratio of False Positive is equal to $5/655 \simeq 1\%$, while the ratio of False Negative is of $6/10 \simeq 60\%$ if we consider the whole set of irregular transactions (first fraudster to mules and mules to last fraudster), or $1/5 \simeq 20\%$ if we consider the subset of irregular transactions detectable by the PSA (mules to last fraudster). For scenario S3, the ratio of False Positive is of $3/5297 \simeq 0.05\%$, while the ratio of False



(a) Real events



(b) Simulated events on scenario S2

Figure 8. Transactions obtained on a simulated run with a limited amount of users. Width of an edge is proportional to the number of transactions and red color corresponds to generated alerts

Negative is of $11/20 \simeq 55\%$ if we consider the whole set of irregular transactions, or $1/20 \simeq 5\%$ if we consider the subset of irregular transactions detected by the PSA.

The fraudulent transactions made from the initial fraudster to the mules are not detected as being suspicious, which is correct as the EPC has not been constructed to detect these transactions. The transactions of all mules, except $EU0$, to the fraudster are correctly detected as being suspicious. All the non-suspicious transactions of mule $EU0$ are detected as being suspicious.

C. Discussion

As we have no ground truth on the real world events, we cannot verify the recognition performance of the PSA. However we can notice that the number of alarms is quite important, and obviously superior to the real quantity of suspicious transactions. In order to reduce the number of alarms, it would be necessary to fine-tune the EPC with the help of a real world dataset with annotated transactions (suspicious/not suspicious)

Table II. SENSITIVITY AND SPECIFICITY OF THE PSA ON DETECTION OF ANOMALIES IN TRANSACTIONS

	Fraudulent	**Normal**	*Total*
Alarm raised	0	0	0
Alarm not raised	0	1077	1077
Total	0	1077	1077

(a) Scenario S1

	Fraudulent	**Normal**	*Total*
Alarm raised	5-1	5	9
Alarm not raised	1+5	650	656
Total	10	655	665

(b) Scenario S2

	Fraudulent	**Normal**	*Total*
Alarm raised	10-1	3	12
Alarm not raised	1+10	5294	5305
Total	20	5297	5317

(c) Scenario S3

or to use an additional component which filters the alerts generated by the PSA.

The PSA shows the correct behavior in all scenarios. The detection errors with user $EU0$ come from the fact that the very first transaction of the user is the fraudulent one. The respective process state component is then set to this amount. Thus all his next transactions are detected as being suspicious as there is no transition in the process behavior representation to the state related to this amount. As a matter of fact, the other transactions of this user will be indefinitely detected as being suspicious in the future.

Usually, the evaluation of anomaly detection tools is done using a ROC curve [18] (sensitivity and specificity obtained for various configuration thresholds $\tau \in \mathbb{R}$). The PSA cannot be configured with such a simple threshold. Instead, there is a complex configuration ($\rho = (\rho_{EPC}, \rho_{mapping})$) composed of an EPC configuration ($\rho_{EPC} \in \mathbb{E}$, $\mathbb{E}$ is the set of possible EPCs) associated with a discretization scheme for transfer amounts ($\rho_{mapping} \in \mathbb{M}$, $\mathbb{M}$ is the set of possible mapping functions). It is thus difficult to automatically run through the possible choices of EPCs and discretization schemes in order to obtain several configurations giving the ROC curve. For this reason, we provide only one performance point.

VI. RELATED WORK

With respect to the exhaustive survey of approaches in the field of business process management given in [19], the functionality the PSA prototype [4], [5] used in this work could be classified as "check conformance using event data" approach. In this approach, information is used both from the process model and the event data in order to identify deviations of runtime behavior from expected behavior. The trend for this specific aspect of business process management, as presented in [19], shows a growing interest in the last three years. A similar approach is described in [20] but the focus is on quantification of inconsistencies by the formation of metrics. We consider the framework presented in [15] on runtime compliance verification for business processes as complementary to our work.

Many data-mining algorithms have been adapted for fraud detection in the banking field. Filters, decision trees and logistic

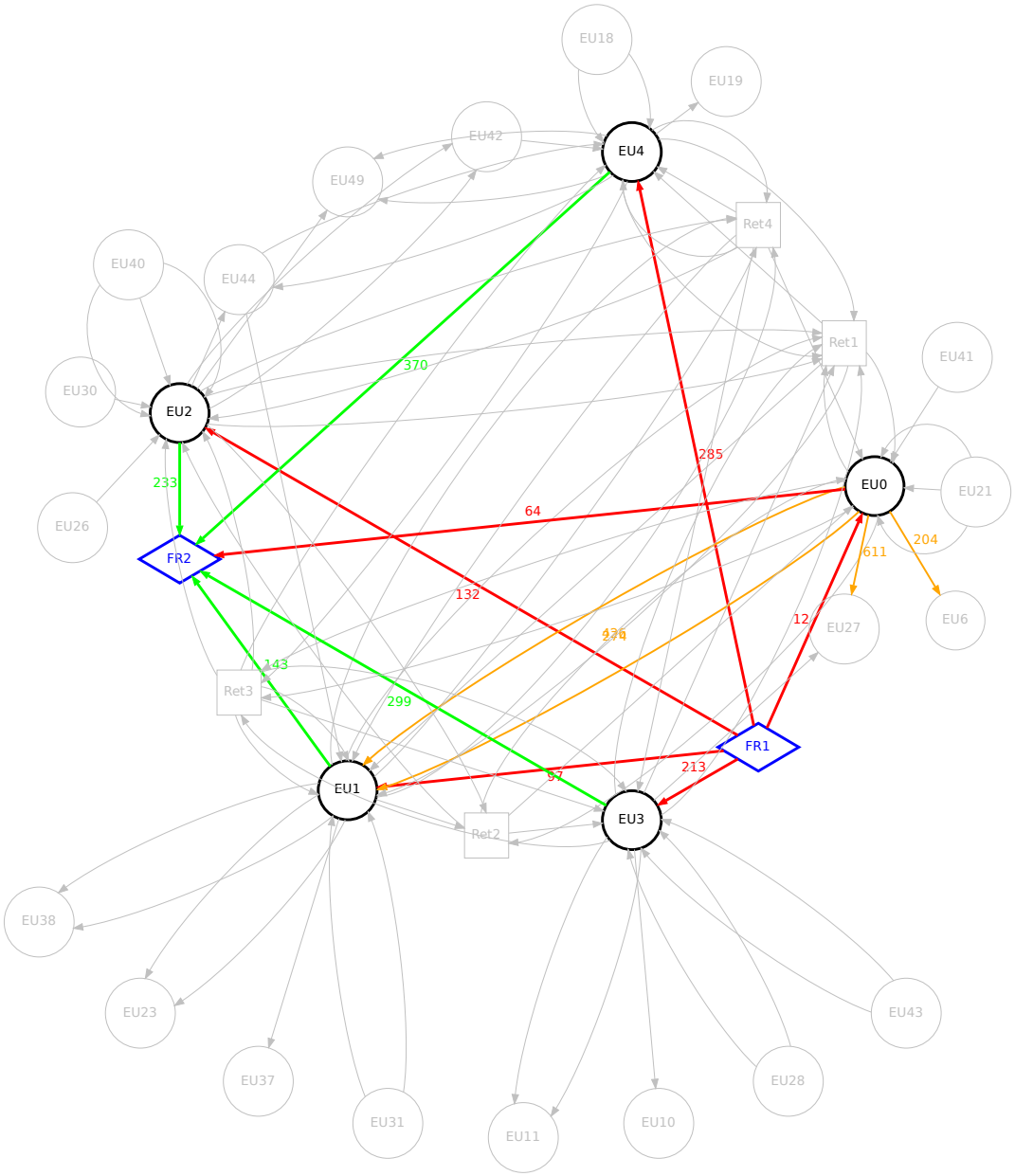


Figure 9. Representation of the users involved in the fraud and their transactions in scenario S2. The transfers of the mules, except those of $EU0$, have been detected. Various transfers from $EU0$ raised false alarms

regression are the most used because they can be easily interpreted. As a result, it is easier for an operator to explain to a client why a specific transaction is considered as being fraudulent. Other methods involving automated model learning are more rarely used because of the difficulty to interpret results and of the need for training data. However, there are some industrial solutions based on such methods. VISA, for example, implements neural networks in their fraud detection tool, RST (Real-Time Scoring) [21]. This tool associates a score to a transaction and raises an alert if the score exceeds a threshold chosen by the bank. However, it is the bank's responsibility to find out the reasons why a transaction which raised an alert should be blocked.

Bhattacharya et.al. [22] and Delamaire et.al. [23] published a state-of-the-art of the data-mining algorithms used for detecting frauds among credit card transactions. They show that several attempts were undertaken to adapt neural networks, SVMs, Bayesian networks, decision trees, expert systems and Hidden Markov Models to the field of credit card transactions. In [24], a separate change detection model for each cell in a multi-dimensional data cube is used for a change detection system for VISA. To our knowledge, not all mobile payment services include automated fraud detection solutions. The surveillance can be manual or based on business rules. However,

the M-PESA service, which is one of the most well known MMT services, has deployed MinotaurTM Fraud Management Solution in 2012 [25]. This fraud management system is based on the use of business rules and neural networks [26]. To our knowledge, there are no public works concerning the study and the adaptation of fraud detection methods to mobile payment systems. Therefore, we cannot easily compare our work to existing systems.

VII. Conclusion

The work presented in this paper utilizes alerts generated by the uncertainty reasoning component of the PSA to detect money laundering patterns in synthetic process behavior composed of simulated logs based on properties captured from real world transaction events.

We have shown that the PSA is able to raise alerts in a simulated scenario of fraud with mules. For this simulated scenario, the detection is efficient, but show that such system could be sensitive to noise in a real world system. It would be necessary to improve the resistance to noise through a correlation of the generated alerts or by an application of specific evaluation of the process states when an alert is generated (for example, move to the critical state if the same alert has been raised several times).

Results of the PSA should be associated with decision and reaction systems in order to modify the security rules of the MMT system to automatically block the fraud [3]. In order to ease the evaluation of the system, it could be interesting to develop methods able to automatically produce a huge quantity of EPCs to provide several evaluation points.

Acknowledgment

The presented work was developed in context of the project MASSIF (ID 257475) being co-funded by the European Commission within the Seventh Framework Programme.

References

[1] CCK, "Quarterly sector statistics report," Communications Commission of Kenya, Tech. Rep., 2012.

[2] Orange, "Orange money," http://www.orange.com/en/press/press-releases/press-releases-2012/Orange-Money-reaches-4-million-customers-and-launches-in-Jordan-and-Mauritius, June 2012, last visit on 12/04/2013.

[3] R. Rieke, L. Coppolino, A. Hutchison, E. Prieto, and C. Gaber, "Security and reliability requirements for advanced security event management," in *Computer Network Security*, ser. LNCS, I. Kotenko and V. Skormin, Eds., 2012, vol. 7531, pp. 171–180.

[4] R. Rieke and Z. Stoynova, "Predictive security analysis for event-driven processes," in *Computer Network Security*, ser. LNCS. Springer, 2010, vol. 6258, pp. 321–328.

[5] J. Eichler and R. Rieke, "Model-based Situational Security Analysis," in *Workshop on Models@run.time*. CEUR, 2011, vol. 794, pp. 25–36.

[6] G. Keller, M. Nüttgens, and A.-W. Scheer, "Semantische Prozeß-modellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)"," *Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Universität des Saarlandes*, vol. 89, 1992.

[7] R. Rieke, E. Prieto, R. Diaz, H. Debar, and A. Hutchison, "Challenges for advanced security monitoring – the MASSIF project," in *Trust, Privacy and Security in Digital Business*, ser. LNCS, S. Fischer-Hübner, S. Katsikas, and G. Quirchmayr, Eds. Springer, 2012, vol. 7449, pp. 222–223. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-32287-7_23

[8] M. Achemlal, S. Gharout, C. Gaber, M. Llanes, E. Prieto, R. Diaz, L. Coppolino, A. Sergio, R. Cristaldi, A. Hutchison, and K. Dennie, "Scenario requirements," MASSIF FP7-257475, Tech. Rep., 2011.

[9] W. Jack, S. Tavneet, and R. Townsend, "Monetary theory and electronic money: Reflections on the kenyan experience," *Economic Quarterly*, no. 96, First Quarter 2010 2010.

[10] FINTRAC Typologies and Trends Reports, "Money laundering and terrorist financing trends in fintrac cases disclosed between 2007 and 2011," http://www.fintrac-canafe.gc.ca/publications/typologies/2012-04-eng.asp#s1-1, April 2012, last visit on 21/05/2013.

[11] Internal Revenue Service (IRS), "Examples of money laundering investigations - fiscal year 2012," http://www.irs.gov/uac/Examples-of-Money-Laundering-Investigations-Fiscal-Year-2012, October 2012, last visit on 21/05/2013.

[12] P. Ochsenschläger, J. Repp, R. Rieke, and U. Nitsche, "The sh-verification tool – abstraction-based verification of co-operating systems," *Formal Aspects of Computing, The International Journal of Formal Method*, vol. 10, pp. 381–404, 1998. [Online]. Available: http://sit.sit.fraunhofer.de/smv/publications/download/FormAsp.ps

[13] W. M. P. van der Aalst, "Formalization and verification of event-driven process chains," *Information & Software Technology*, vol. 41, no. 10, pp. 639–650, 1999.

[14] M. Weber and E. Kindler, "The petri net markup language," in *Petri Net Technology for Communication-Based Systems*, ser. LNCS. Springer, 2003, vol. 2472, pp. 124–144.

[15] F. M. Maggi, M. Montali, M. Westergaard, and W. M. P. van der Aalst, "Monitoring business constraints with linear temporal logic: An approach based on colored automata," in *Business Process Management (BPM 2011)*, ser. LNCS, vol. 6896. Springer, 2011, pp. 132–147.

[16] R. Rieke, R. Giot, and C. Gaber, "Predictive security analysis - concepts, implementation, first results in industrial scenario," 2013, talk at CYBER SECURITY & PRIVACY EU FORUM 2013. [Online]. Available: http://www.cspforum.eu/uploads/Presentation-Roland_Rieke.pdf

[17] C. Gaber, B. Hemery, M. Achemlal, M. Pasquet, and P. Urien, "Synthetic logs generator for fraud detection in mobile transfer services," in *Proceedings of the 2013 International Conference on Collaboration Technologies and Systems (CTS2013)*, 2013.

[18] T. Fawcett, "Roc graphs: Notes and practical considerations for researchers," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 882–891, 2004.

[19] W. M. P. van der Aalst, "Business process management: A comprehensive survey," *ISRN Software Engineering*, p. 37, 2013.

[20] A. Rozinat and W. van der Aalst, "Conformance checking of processes based on monitoring real behavior," *Information Systems*, vol. 33, no. 1, pp. 64 – 95, 2008.

[21] VISA, "Security and trust at every level," http://www.visaeurope.com/en/about_us/security.aspx, last visit on 22/03/2013.

[22] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, 2011.

[23] L. Delamaire, H. Abdou, and J. Pointon, "Credti card fraud and detection techniques : a review," *Banks and Bank systems*, vol. 4, 2009.

[24] C. Curry, R. L. Grossman, D. Locke, S. Vejcik, and J. Bugajski, "Detecting changes in large data sets of payment card data: a case study," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '07, 2007, pp. 1018–1022.

[25] E. Okutyi, "Safaricom tightens security on m-pesa with fraud management system," http://www.humanipo.com/news/1341/Safaricom-tightens-security-on-M-Pesa-with-Fraud-Management-system, August 2012, last visited on 22/03/2013.

[26] Neural technologies, "Minotaurtm fraud detection software - finance sector," http://www.neuralt.com/fraud_detection_software.html, last visited on 23/03/2013.