From Fraud Vulnerabilities and Threats to Fraud Avoidance and Tolerance

Lilien, Leszek; Bhargava, Akhil; Bhargava, Bharat

Abstract—*Fraud vulnerability and fraud threat assessments can improve the level of trustworthiness in computer systems by reducing fraud attacks. A deep knowledge of fraud vulnerabilities and fraud threats can be utilized to avoid/tolerate fraud attacks. A swindler is a legitimate user who intentionally benefits from the system or other users by deception. We describe an architecture for swindler detection consisting of four components. One of them runs the deceiving intention predictor algorithm to foresee fraudulent intentions. It is effective in uncovering different fraud attack strategies, from naive to smart ones*

Index Terms—*Computer security, fraud, threats, trusted computing, vulnerabilities*

1 INTRODUCTION

EVER-GROWING dependence of our civilization on ubiquitous computer systems brings with it an increase in vulnerabilities and threats, including fraud vulnerabilities and fraud threats, experienced by individuals, systems, and enterprises. The need to ameliorate these negative side effects of information technology, ranging from accidental failures to terrorist security attacks, becomes more urgent every day.

1.1 Basic Ideas and Terminology

Vulnerability is defined as a flaw or weakness in system security procedures, design, implementation or internal controls that could be exercised (accidentally triggered or intentionally exploited), and results in a penetration: a security breach or a violation of the security policy [34].

We define *threats* against systems as entities that can cause security breaches [20, 34]. An *attack* is an intentional exploitation of vulnerabilities, and an *accident* is an inadvertent triggering of vulnerabilities. A threat by itself causes no harm. To cause harm, its potentiality must *materialize* as an attack or an accident.

Trust can be defined as a firm reliance on the integrity, ability, or character of a person or thing [36]. This definition extends trust to things.

Fraud (a *fraud attack*) is a deception deliberately practiced in order to secure unfair or unlawful gain [36], or an intentional perversion of truth in order to induce another to part with something of value or to surrender a legal right [25]. *Fraud attacks* are a special class of security attacks, defined by attacker's goal of gaining a financial or other tangible reward.

Entities (human or artificial) that commit fraud are known as *fraudsters*. They can be classified into two categories: impersonators and swindlers [11]. An *impersonator* is an *illegitimate* user who steals resources from victims, for instance by taking over their accounts. A *swindler* is a *legitimate* user who intentionally benefits from the system or other users by deception. For instance, she obtains legitimate accounts and uses the services without intention to pay the bills.

We extend the notion of a fraudster to cover not only humans but also artificial entities (such as programs, *computers*, etc.).

1.2 Related Work on Fraud

Fraud detection systems are widely used in telecommunication, online transactions, computer and network security, and insurance. A taxonomy of computer fraud is

Manuscript received on November 17, 2007, accepted on March 15, 2009.

L. Lilien is with the Department of Computer Science, Western Michigan University, Kalamazoo, MI, USA. A. Bhargava is with the School of Electrical and Computer Engineering; and B. Bhargava is with the Department of Computer Sciences, Purdue University, West Lafayette, IN, USA

presented by Vasius [37]. Many research efforts address fraud detection in telecommunications (e.g., [2, 13, 35]). Examples of other common fraud research address identity fraud [19], and fraud in e-commerce and e-business [29, 24, 38]. Data mining [1, 12, 16], machine learning [14, 27], and statistical methods [13, 35] have been developed to detect fraud.

Different fraud detection approaches include an adaptive rule-based detection framework for superimposition fraud [16], a neural network technique [13], solving the incompatible schema problem caused by sharing a distributed fraud-detection database [15].

Due to the skewed distribution of fraud, one challenge in fraud detection is a very high *false alarm* rate. Also several other criteria are used to evaluate performance of fraud detection engines. *Receiver Operating Characteristics (ROC)* [17, 25, 35] is a common one. Rosset *et al.* [31] use accuracy and fraud coverage as criteria.

A cost-based metric can be used in commercial fraud detection systems [33].

Security techniques such as cryptography, separation of duty policies, biometric authentication, and state transition analysis can prevent fraud [21]. Also general techniques are applicable, e.g., use of adaptability [9, 16, 23].

Impersonators can be forestalled by utilizing *cryptographic techniques* that provide strong protection to users' authentication information.

The idea of *separation of duty* [20] may be applied to reduce the impact of a fraudster. An example of this idea is the *Chinese wall policy* arising in the commercial sector of consulting services is an example [11].

For more on related work cf. Reference [6].

2 Towards Trusted Computing

In this and the following sections, we discuss one approach for reaching the goal of trusted computing. We proceed from analysis of fraud vulnerabilities, via fraud threats, and via providing fraud attacks to establishing an environment of mutual trust among human and technological entities involved in computing, thus providing a *trustworthy* computing environment.

We concentrate on the integrity aspects of security (pretty much ignoring the other two elements of classically defined security: availability and confidentiality) [18, 30].

3 Fundamental Principles

We start by presenting the fundamental principles of our approach. We propose that the following principles be employed for creating integrity-improving algorithms or procedures, methods and methodologies:

1. Apply research principles from the areas of reliability, integrity and fault tolerance [5, 8, 7].
2. Apply non-deterministic system behavior, with the ideas of uncertainty, unpredict-ability, and deception. E.g., use honeypots [26, 32].
3. Employ adaptability, diversity, flexibility [4, 9], so a system can adaptively decide which of the alternative executables it runs.
4. Use appropriate military principles for defensive or even offensive actions [22]. Among the candidate principles are: defense in depth; use of spies, deception, surprise; and counter-attacking (when allowed by law).
5. Apply biologically-inspired principles, such as autonomic computing [3], including self-configuring, self-healing, self-optimizing, and self-protecting.

4 Analyzing Fraud Vulnerabilities

4.1 Fraud and Trust

Fraud involves *abuse of trust* [10, 39]. A fraudster strives to present himself as a trustworthy individual and friend. In a clear way, the more trust one places in others, the more open or even careless one tends to become in relationships with them. This results in becoming much more vulnerable (with the possibility of experiencing more harm). Fortunately, vulnerabilities are not automatically exploited but create only a *potential* for fraud. Only misplaced trust may end up being exploited.

4.2 The Nature of Fraud Vulnerabilities

Vulnerabilities create an *opportunity* for fraud. E.g., if Bob learns that Alice has a new account but is not using it, he can break in and use it.

While stealing is a singular act, fraud might be a *series* of acts being repeated over and over again as long as the victim is *unaware* of these acts or of their perpetrator.

Summarizing, fraud has *opportunistic*, *serial* and *covert* nature. (The serial nature encourages to model fraud as *nested transactions* [28].)

4.3 Categorization of Fraud Vulnerabilities

Fraud vulnerabilities can lead to recurring attacks and increasing damage in the enterprise. They appear during the design, implementation, and deployment stages of a system's lifetime.

Taxonomies of fraud vulnerabilities for each stage facilitate gaining a deep insight into the realm of vulnerabilities. The taxonomy can be used by a formal model to reason about characteristics of vulnerabilities, and to eliminate some of them. Good fraud vulnerability taxonomy can guide system design and implementation, and facilitate avoiding and tolerating fraud attacks in deployed systems. In particular, a proper vulnerability categorization underlies analytical and experimental methods for assessing the system-wide impact of vulnerabilities.

Causes of fraud vulnerabilities include the following categories of causes: (a) location-related, (b) rush-related, (c) mobility-related, and (d) software-related (cf. [6] for more details).

4.4 Elimination of Fraud Vulnerabilities

Elimination of fraud vulnerabilities is an effective way of dealing with fraud. However, it is *not* necessarily the most efficient way. Hence, we may need to accept even some known fraud vulnerabilities and instead try to ameliorate them by analyzing potential threats that could exploit them. This is the topic of the next section.

5 Analyzing Fraud Threats

Efficient methods to assess threats resulting from system flaws allow for analysis of all kinds of threats, as well as evaluation of their potential impacts. The assessment precedes building a robust threat avoidance/tolerance mechanism for handling fraud threats due to known or even unknown fraud vulnerabilities.

Also in this case proper fraud threat categorization facilitates the analysis.

5.1 Fraud Threats as Subcategory of Security Threats

Fraud threats can be viewed as a special subcategory of general security or privacy threats that have a few salient features. First, a fraudster is usually *well known* to the victim whose trust he first gains and then abuses. In contrast, a "generic" attacker is usually not as closely related to a victimized system as a fraudster to his victim. This is true even in case of perpetrators of "generic" insider attacks.

Second, the goal of a fraudster is to *benefit himself* and not to hurt the system. Unlike a malicious attacker, fraudster does not plan to disrupt or destroy the system that he exploits. This is similar to the relationship between a clever parasite and the organism on which it feeds. Negative consequences of fraudster's actions are only unavoidable side effects.

Third, once a suspicion of a fraud threat arises and becomes known to the fraudster who is still not identified, the fraudster usually moves to prey on another victim. In contrast, when a suspicion of a malicious attack threat arises and becomes known to the attacker, the threat does not vanish: the attacker will probably just use different venues of attack without changing its object.

5.2 Fraud Attacks as Subcategory of Security Attacks

Similarly, materialized fraud threats, that is fraud attacks, can be viewed as a special subcategory of general security or privacy attacks, distinguished by a few salient features. First, fraud is often committed *over a period of time*, if not detected, even over months or years. Many other security attacks occur over a relatively short time

span even though attack preparation can also be surprisingly patient and slow.

Second, a successful fraud requires *keeping secrecy*. The longer it is kept the better for the fraudster, who can repeat or escalate his fraud. In contrast, an attacker often wants her exploits to be known to the general public (for fame, intimidation, etc.). Even if she does not want it, due to very visible impacts of her exploits, she often cannot conceal the attack effects.

5.3 Other Fraud Threat Features and Types

Fraud often occurs as a malicious *opportunistic reaction*, triggered by a careless action. In other words, quite often people do not plan fraud but commit it when tempted by a vulnerability revealed to them. When the vulnerabilities become known to recidivist fraudsters, the risk of becoming a fraud victim rises sharply.

Fraud escalation seems to be a natural phenomenon. A fraudster, maybe reacting to an opportunity, defrauds five dollars. Next time, encouraged by the ease of the act, he might defraud a much more significant amount.

Gang fraud threats can be especially damaging since they involve a group of collaborating fraudsters (cf. [6] for more details).

There exist environments contributing to fraud threats, such as the ones with fuzzy assignment of responsibilities between participating entities, be they human or artificial (cf. [6]).

5.4 Need for Fraud Threat Assessment

Fraud threats resulting from system flaws should be assessed at each stage of system lifetime by efficient methods. A good taxonomy facilitates this analysis and further investigation of threats. It is important to notice that a comprehensive assessment of fraud threats may identify even unknown fraud vulnerabilities, not identified by fraud vulnerability analysis.

A threat analysis precedes developing robust threat avoidance/tolerance methods, mechanisms, and tools. Designers and implementers of computer systems need them at the development time stages of system life cycle. Similarly, system administrators and maintenance staff need them at the post-deployment time stages (after the system is deployed).

6 Fraud Threat Avoidance and Tolerance

Detection of fraud *vulnerabilities* and fraud *threats* is a prerequisite for or a component of fraud avoidance and tolerance. In addition, detection of fraud *attacks* might be a prerequisite for or a component of fraud tolerance (but not for fraud avoidance since preventing fraud attacks is assumed for such an approach).

6.1 Fraud Threat Avoidance

The problem of threat *avoidance* is most important at the development time for computer systems. The situation is exacerbated by the inherent fraudsters' advantage: once a typical system is deployed, its overall structure and its many functionalities are pretty much frozen (at least until the next major system release). Fraudsters have a lot of time to study and probe the system to discover its vulnerabilities, and then to exploit them. Unless the system has built-in capabilities for fault and intrusion tolerance, the only weapons in developers' hands are security patches and point releases fixing bugs (non-structural and of limited scope).

The designers should therefore be very careful what they let out of their door. Yet, too often designs of even critical systems consider threat avoidance to an insufficient degree, making them just too vulnerable to even relatively unsophisticated attacks.

The promising known fraud prevention techniques include cryptography, separation of duty policies, and state transition approaches.

6.2 Fraud Threat Tolerance

The goals for fraud threat *tolerance* are analogous. In addition to arming the system for threat tolerance at the development time, good algorithms and tools must be provided for post-deployment support. This includes the techniques analogous to the ones used for fault tolerance in the field of computer reliability.

6.3 Post-deployment Detection of Fraud Vulnerabilities and Threats

In addition to fraud vulnerability and threat analysis done at the system development time, we must ensure that deployed systems are sub-jected to detection *of fraud* vulnerabilities and threats also after deployment. This can be done by system testing, analogous to reliability testing.

Costs of testing should be weighed against the expected fraud losses and future benefits, since testing should enhance future system performance. Unfortunately, irrationally extensive fraud protection measures might be an irrational—albeit psychologically explainable—reaction by an entity just recently defrauded.

Exhaustive fraud vulnerability or threat testing might not be a practical approach, so sampling could be used (cf. [6] for more details).

6.4 Detection of Fraud Attacks

After system deployment, in addition to identifying fraud *vulnerabilities* or fraud *threats* (discussed above), fraud threat tolerance may require tests for detecting fraud *attacks*.

The two main approaches for fraud detection are: profile-based anomaly detection and rule-based fraud detection.

7 An Architecture for Swindler Detection

In the preceding sections we proposed an approach to analyzing fraud vulnerabilities, fraud threats and fraud attacks. In this and the next section, we present a system for swindler detection, and an analysis of its critical algorithm.

The major challenge for swindler detection is to react to a suspicious action or cooperation that may lead to a fraud. Three approaches were considered: (1) detecting an entity's activities that deviate from legitimate patterns; (2) constructing state transition graphs for existing fraud scenarios and detecting frauds similar to the known ones; and (3) discovering an entity's intention based on its past behaviors. The first two approaches are also used to detect frauds conducted by impersonators. The last one is applicable only for swindler detection. An architecture utilizing all three approaches, briefly described in the next section, was proposed in Reference [11].

7.1 Design Considerations and Top-level Architecture

The design of the architecture was based on the following assumptions:

- A deviation from the usual pattern of an entity behavior may indicate a fraud.
- A similarity between an entity's current activity and a known fraud scenario warns that the same fraud may be occurring again.
- Analysis of an entity's behaviors in a relatively long period may reveal entity's bad intentions that it tries to mask by blameless activities.

Our swindler detection architecture consists of four components:

1. Profile-based *anomaly detector* monitors current activities for suspicious actions based upon the established patterns of an entity's behavior. It outputs the *fraud confidence* indicator showing the probability of a fraud.
2. *State transition analysis* component builds for current activities a state transition graph that provides so called *state descriptions* when a current activity results in entering a danger state, which may lead to a fraud.
3. *Deceiving intention (DI) predictor* discovers deceiving intention of an entity based on entity's history and *satisfaction ratings,* that is, in contrast to the previous two components it investigates only entity's past behaviors. *DI-confidence* indicator is a measure that characterizes the belief that the target entity has a deceiving intention. It is a real number ranging over [0,1], with the higher values indicating stronger beliefs.
4. *Decision-making* component takes as its inputs the outputs of the three preceding components, namely, fraud confidence, state description, and DI-confidence. It assists system administrators in reaching fraud alarm decisions, based on the predefined policies.

7.2 Anomaly Detector Component

Profile-based anomaly detector monitors for a target entity's activities that deviate from established patterns. It consists of three major subcomponents:

1. The *rule generation and weighing* subcomponent applies data mining techniques to existing massive amounts of entity activity records. From this information, fraud rules are generated and assigned weights according to their frequency of occurrence. Both entity and behavior attributes are used in mining and weighing fraud rules.
2. The *user profiling* subcomponent produces the profiling information characterizing entities, such as age, location and financial status. It also captures an entity's behavior patterns, such as how often she buys or sells, the price preferences and product choices.

   There are two sets of profiling data, one for *current profiles* and the other for *historical profiles*. In order to reflect an entity's current behavior patterns, the current profile set is dynamically updated according to behaviors. As behavior data grows larger, the decay process is used to reduce the data volume.

   This subcomponent also involves rule selection for a specific entity, based on profiling results and rules. When combined with profiling information, a set of rules is selected as fraud indicators for monitoring a specific entity.
3. The *online detection* subcomponent retrieves the related fraud rules when an activity occurs. It may also need to retrieve the entity's current and historical behavior patterns. Each rule is checked and a weight for it is produced.

   If a behavior deviation reaches the defined threshold, the offending entity will be caught. A weight will be output according to the rules. The results are combined to determine fraud detection confidence.

7.3 State Transition Analysis Component

State transition analysis models fraud scenarios as series of states changing from an initial secure state to a final compromised state. The *initial sta-te* precedes actions that lead to a fraud. The *final state* is the resulting state of committing a fraud. There may be several *intermediate states* between them. Actions that cause transition from one state to another are called *signature action*s. They are the smallest actions required to transition closer to the final state. A fraud scenario will not be completed without them.

This model requires collecting fraud scenarios at the beginning and identifies the initial and final states. Then, the signature actions for that scenario are identified in the backward direction. A *fraud scenario* is represented as a state transi-tion graph of states and signature actions.

A *danger factor* is associated with each state. It is defined as the distance from the current state to the final state that indicates a fraud. If one sta-te leads to several final states, the minimum dis-tance is used. For each activity, state transition analysis checks the *potential next states.* If the maximum value of the danger factors associated with these potential next states exceeds a thre-shold, a warning is raised and a state description is sent to the decision-making component.

7.4 Deceiving Intention Predictor Component

The kernel of deceiving intention predictor is the *deceiving intention prediction (DIP) algorithm* [11] that views a belief in a deceiving intention as complementary to a trust belief. The trust belief for an entity is evaluated based on the satisfaction sequence $< S_1, S_2, \ldots, S_n>$. S_n is the most recent satisfaction rating, which contributes the α portion to the trust belief. The remaining portion, $(1 - \alpha)$, comes from the previous trust belief and is determined recursively.

For each entity, DIP maintains a pair of factors: the *construction factor* W_c and the *destruction factor* W_d. If integrating current satisfaction rating increases trust belief, then $\alpha = W_c$; otherwise $\alpha = W_d$. W_c and W_d are initialized by the system administrator. They satisfy the constraint $W_c < W_d$ so that the property of *easy-destruction-hard-construction* is assured. This property stems

from the fact that more effort is needed to gain the same amount of trust than to lose it.

W_c and W_d are modified when a foul event occurs. A *foul event* is triggered when a satisfaction rating is lower than the established threshold. Upon a foul event, the target entity is put under supervision in the sense that entity's W_c is decreased and W_d is increased.

If the entity does not conduct any foul event during a *supervision period,* entity's W_c and W_d are restored to the initial values. Otherwise, entity's W_c and W_d are further decreased and increased, respectively. The supervision period associated with an entity increases each time when the entity is put under supervision, so that this punishment lasts longer the next time. In this way, an entity with a worse history is treated harsher. The *DI-confidence* is computed as *1 – current_trust_belief.*

7.5 Decision-making Component

The decision-making component takes fraud confidence, state description, and DI-confidence as its inputs. It passes warnings from state transition analysis to system administrators, and displays the description of a next potential state in a readable format. It computes the *expected fraud risk*, raising a fraud alarm when this risk ex-ceeds the corresponding fraud investigation cost.

8 Experimental Study of DIP Algorithm for Deceiving Intention Predictor

8.1 Three Types of Deceiving Behaviors

Experimental evaluation of the DIP algorithm [11] investigates its performance for three types of deceiving behavior identified by us:

1. Behavior with *uncovered deceiving intentions,* where swindler's satisfaction ratings are stably low and vary in a small range over time.
2. Behavior with *trapping intentions,* where a swindler first exhibits intentionally blameless behavior to gain trustworthiness. This is only a preparation for a fraud, which follows.
3. Behavior with *illusive intentions,* where a swindler exhibits cycles of intentionally blameless behavior followed by intervals fraudulent actions. The periods of apparently blameless behavior are meant to cover the dishonest activities and to confuse the fraud detection mechanisms. This results in *cycles* of preparation and entrapment (in contrast to the previous case when one preparation interval precedes one entrapment period).

Representations of these behaviors are inputs to the DIP algorithm, which calculates for them the value of the DI-confidence indicator (which is a real number ranging over [0,1] with the higher values indicating stronger beliefs).

8.2 Results

The experimental results can be summarized as follows [11]:

1. For a *swindler with uncovered deceiving intentions:* Since the possibility that the swindler conducts foul events is high, he is under supervision most of the time. The construction and destruction factors become close to 0 and 1, respectively, due to repetitive punishment for foul events. The trust values at 0.1 are close to the minimum. The DI-confidence is around 0.9.
2. For a *swindler with trapping intentions:* The DIP algorithm responds quickly to the sharp drop of the satisfaction rating when swindler ends the preparation phase and enters the entrapment phase. Increasing DI-confidence from 0.22 to 0.76 takes only 6 ratings.
3. For a s*windler with illusive intentions*: DI-confidence increases when the swindler ends the preparation phase of a cycle and starts an entrapment. Similarly, DI-confidence decreases when the swindler ends the entrapment phase and enters the "honest" phase of a cycle. Still, the DIP algorithm is able to catch this smart swindler because her DI-confidence eventually increases to about 0.9. This demonstrates that an effort to cover periods of fraudulent activities with periods of good behaviors is less and less effective with each repetition of the preparation-entrapment cycle.

9 CONCLUSION

We discussed relationships between fraud vulnerabilities, fraud threats, fraud attacks, and trust. We analyzed fraud, identifying the salient features of fraud attacks, which make them a separate category within the realm of security attacks. We proposed architecture for swindler detection consisting of four components: profile-based anomaly detector, state transition analysis, deceiving intention predictor, and decision-making component. The deceiving intention predictor (DIP) algorithm is the critical element of the architecture. The DIP algorithm predicts fraudulent intentions. Its experimental evaluation shows that it is effective in uncovering different fraud strategies, from naive to smart ones.

Acknowledgements

This research was supported in part by the NSF grants IIS-0242840 and IIS-0209059.

REFERENCES

[1] D. W. Abbott, I. P. Matkovsky, and J. F. Elder, "An evaluation of high-end data mining tools for fraud detection," *Proc. 1998 IEEE Intl. Conf. on Systems, Man, and Cybernetics*, 1998.

[2] R. Alves, P. Ferreira, O. Belo and J. Lopes, "Discovering telecom fraud situations through mining anomalous behavior patterns," *Proc. DMBA Workshop, 12th ACM SIGKDD*, 2006.

[3] Autonomic Computing, *IBM Systems J*. (special issue), Vol. 42(1), 2003.

[4] C. Bain, D. B. Faatz, A. Fayad, D. Williams, "Diversity as a defense strategy in information systems. Does evidence from previous events support such an approach?" *Proc. IICIS 2001*.

[5] Anjali Bhargava and B. Bhargava, "Applying fault-tolerance principles to security research," *Proc. IEEE Symposium on Reliable Distributed Systems*, New Orleans, LA, Oct. 2001.

[6] B. Bhargava, "Vulnerabilities and Fraud in Computing Systems," *Proc. International Conf. on Internet, Processing, Systems, Interdisciplinaries* (*IPSI*), *VIP Scientific Forum*, Sv. Stefan, Serbia and Montenegro, Oct. 2003.

[7] B. Bhargava and L. Lilien, "A Review of Concurrency and Reliability Issues in Distributed Database Systems," pp. 1-84 in: *Concurrency Control and Reliability in Distributed Systems,* ed. B. Bhargava, Van Nostrand Reinhold, New York, New York, 1987.

[8] B. Bhargava and L. Lilien, "Expert Systems for Fault Tolerant Distributed Database Systems," pp. 41-182 in: *Essays in Computer Vision and Other Topics*, ed. J. Tou, Academia Sinica, Republic of China, 1990.

[9] B. Bhargava and J. Riedl, "A Formal Model for Adaptable Systems for Transaction Processing," *IEEE Transactions on Knowledge and Data Engineering,* Vol. 4, No. 1, 1989.

[10] B. Bhargava and Y. Zhong, "Authorization Based on Evidence and Trust," *Proc 4th Intl. Conf. on Data Warehousing and Knowledge Discovery (DaWaK-2002),* Aix-en-Provence, France, Sep. 2002.

[11] B. Bhargava, Y. Zhong, and Y. Lu, "Fraud Formalization and Detection," *Proc. 5th Intl. Conf. on Data Warehousing and Knowledge Discovery (DaWaK-2003),* Prague, Czechia, Sep. 2003.

[12] R. Brause, T. Langsdorf, and M. Hepp, "Neural data mining for credit card fraud detection," *Proc. 11th IEEE Intl. Conf. on Tools with Artificial Intelligence*, 1999.

[13] P. Burge and J. Shawe-Taylor, "Detecting cellular fraud using adaptive prototypes," *Proc. AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Mgmt*, 1997.

[14] P. Burge, J. Shawe-Taylor, Y. Moreau, B. Preneel, C. Stoermann, and C. Cooke, "Fraud detection and management in mobile telecommunications networks," *Proc. European Conf. on Security and Detection*, 1997.

[15] P. Chan, W. Fan, A. Prodromidis, and S. Stolfo, "Distributed data mining in credit card fraud detection," *IEEE Intelligent Systems*, 1999.

[16] T. Fawcett and F. Provost, "Adaptive Fraud Detection," *Data Mining and Knowledge Discovery,* Vol. 1(3), 1997.

[17] J. Hollmén, J. and V. Tresp, "Call-based fraud detection in mobile communication networks using a hierarchical regime-switching model," *Advances in Neural Information Processing Systems 11: Proc. 1998 Conf.*, 1998.

[18] Information Technology Security Evaluation Criteria (IT-SEC): Provisional Harmonized Criteria, Dec. 1993.

[19] R. Jamieson, D. Winchester, G. Stephens, S. Smith, "Developing a Conceptual Framework for Identity Fraud Profiling," *Proc. European Conf. on Information Systems*, Galway, Ireland, 2008.

[20] E. Jonsson, L. Strömberg, and S. Lindskog, "On the Functional Relation between Security and Dependability Impairments," *Proc. Workshop on New Security Paradigm*s, Sep. 1999.

[21] E. Kotsakis *et al.*, "Advancing Security & Anti-fraud by Means of Info-mobility and Navigation Technologies (SAINT). System Preliminary Design," European Commission, June 2001.

[22] R. R. Leonhard, *The Principles of War for the Information Age,* Presidio Press, Novato, CA, 1998.

[23] L. Lilien and Anu Bhargava, "From Vulnerabilities to Trust: A Road to Trusted Computing," *Proc. International Conf. on Internet, Processing, Systems, Interdisciplinaries* (*IPSI*), *VIP Scientific Forum*, Sv. Stefan, Serbia and Montenegro, October 2003.

[24] I. MacInnes, D. Musgrave, and J. Laska, "Electronic Commerce Fraud: Towards an Understanding of the Phenomenon," *Proc. 38th Hawaii Intl. Conf. on System Sciences*, 2005.

[25] *Merriam-Webster's Collegiate Dictionary*, *Eleventh Edition*, 2003.

[26] D. B. Moran, "Trapping and Tracking Hackers: Collective security for survival in the Internet Age," *Proc. Third Information Survivability Workshop,* Boston, MA, Oct. 2000.

[27] Y. Moreau, H. Verrelst and J. Vandewalle, "Detection of mobile phone fraud using supervised neural networks: a first prototype," *Proc. Intl. Conf. on Artificial Neural Networks*, 1997.

[28] J. E. B. Moss, "Nested transactions: an approach to reliable distributed computing," Ph.D. Thesis, MIT, 1981. Available as Technical Report MIT/LCS/TR-260.

[29] S. Pandit, D.H. Chau, S. Wang, and C. Faloutsos, "NetProbe: A Fast and Scalable System for Fraud Detection in Online Auction Networks," *Proc. 16th Intl. Conf. on World Wide Web (WWW'07)*, Banff, Alberta, Canada, May 2007.

[30] C. P. Pfleeger and S.L. Pfleeger, *Security in Computing. Fourth Edition*, Prentice Hall PTR, 2007.

[31] S. Rosset, U. Murad, E. Neumann, Y. Idan, and G. Pinkas. "Discovery of fraud rules for telecommunications - challenges and solutions," *Proc. 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 1999.

[32] L. Spitzner, "Honey pots: Definition and value of honey pots," <http://www.enteract.com/~lspitz/honeypot.html>.

[33] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan, "Cost-based modeling for fraud and intrusion detection: results from the JAM project," *Proc. DARPA Information Survivability Conf. and Exposition*, 2000.

[34] G. Stoneburner, A. Goguen, and A. Feringa, "Risk Management Guide for Information Technology Systems," NIST Special Publication 800-30, Washington, DC, 2001.

[35] M. Taniguchi, M. Haft, J. Hollmén, and V. Tresp, "Fraud detection in communications networks using neural and probabilistic methods," *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, 1998.

[36] *The American Heritage Dictionary of the English Language*, Fourth Edition, Houghton Mifflin, 2000.

[37] L. Vasiu and I. Vasiu, "Dissecting Computer Fraud: From Definitional Issues to a Taxonomy," *Proc. 37th Hawaii Intl. Conf. on System Sciences*, 2004.

[38] L. Vasiu, "A Conceptual Framework of E-Fraud Control in an Integrated Supply Chain," *Proc. European Conf. on Information Systems*, Turku, Finland, 2004.

[39] Y. Zhong, Y. Lu, and B. Bhargava, "Dynamic Trust Production Based on Interaction Sequence," Technical Report CSD-TR 03-006, Department of Computer Sciences, Purdue University, Mar. 2003.