Case Studies of an Insider Framework

Matt Bishop, Sophie Engle,
Sean Peisert, Sean Whalen
University of California, Davis
Davis, CA
{bishop,engle,peisert,whalen}@cs.ucdavis.edu

Carrie Gates
CA Labs
Islandia, NY
carrie.gates@ca.com

Abstract

Much of the literature on insider threat assumes, explicitly or implicitly, a binary, perimeter-based notion of an insider. However, it is generally accepted that this notion is unrealistic. The Attribute-Based Group Access Control (ABGAC) framework is a generalization of Role-Based Access Control (RBAC) which allows us to define a non-binary notion of "insiderness". In this paper, we illustrate how to use ABGAC to perform insider threat analysis of high-risk resources with three case studies. This precise yet flexible identification of high-risk resources and associated insiders allows organizations to understand where to target efforts towards defending against the insider problem.

1 Introduction

The insider problem is considered the most dangerous threat to computer and infrastructure security today, and also the most difficult to remediate. Equally important is the need to identify cost-effective defenses against insider attacks. Most of the literature reflects a binary notion of an insider – either someone falls inside some well-defined perimeter or does not. As a consequence, all insiders are considered equal.

But all insiders are *not* equal, and treating insiders in this way diverts focus away from an organization's critical information resources. Some insiders are more of a threat than others, and hence have a higher degree of "insiderness". However, the traditional binary model of an insider does not lend itself to this type of meaningful threat analysis.

Further complicating the binary definition is the erosion of well-defined perimeters, especially with respect to computer systems [16]. Organizations must provide multiple levels of access to their information resources as they hire contractors, outsource to other organizations, partner and merge, and use software-as-a-service offerings such as SalesForce.com as part of their core business. As a result, the perimeter-based approach to defining insiders is becoming less meaningful, in the same way that traditional perimeter-based computer security defenses such as firewalls and network intrusion detection systems no longer protect against attacks or mistakes occurring on the inside, including Trojan horses, phishing, and client-side, cross-site scripting attacks.

We have previously presented Attribute-Based Group Access Control, which is capable of capturing the non-binary notion of "insiderness" by defining insiders with respect to a resource [4]. Additionally, we have extended ABGAC into a useable framework for analyzing insider threat, using a hierarchy of security policies to categorize where insider problems may arise [3]. In this paper, we take this existing framework and apply it to three case studies, illustrating how to use ABGAC to perform insider threat analysis of high-risk resources.

The rest of this paper is organized as follows. We describe the ABGAC framework in Section 2. We use three case studies to illustrate how to perform insider threat analysis with ABGAC. The first case

study in Section 4 illustrates how ABGAC is able to capture an insider attack predating the widespread adoption of computer systems. We contrast that with the second case study in Section 5, which includes modern attacks against computer systems. The last case study examines the recent San Francisco insider attack involving a withheld administrator password. While these case studies do not capture the wide-array of possible insider attacks, it does illustrate the flexibility of ABGAC. We briefly discuss related work in Section 7. Finally, we provide some concluding remarks in Section 8.

2 Definition

Bishop [2] proposed a definition of an insider as: "a trusted entity that is given the power to violate one or more rules in a given security policy ... the insider threat occurs when a trusted entity abuses that power." This definition suggests an insider must be determined with reference to some set of rules that is part of a security policy. We expanded that definition [4, 3] by arguing that a security policy is inherently represented by the access control rules employed by an organization. So, an insider is defined with regard to two primitive actions:

1. violation of a security policy using legitimate access, and
2. violation of an access control policy by obtaining unauthorized access.

In the first case, the insider uses their legitimate access to perform some action that is contrary to the security policy, such as might be observed when sensitive data is leaked to some third party or when access to a resource is given or blocked. Here the insider has legitimate access to the data or resources, but uses that access to provide the information to someone who does not themselves have access (or to deny access to someone who does have access). In the second case, the insider uses their access to extend their privileges in a manner that breaks both the access control and security policies. An example of such a breach occurs when a user might have a legitimate capability to log into a particular system, but then abuses that privilege to gain illegitimate superuser-level access to the system (e.g., by exploiting some system vulnerability such as a buffer overflow or race condition).

As we discussed earlier, previous definitions gave rules or descriptions intended to allow the reader to determine who is an insider, resulting in a binary distinction: an entity is either *an insider* or *not an insider*. We argue that a non-binary approach is required, to indicate degrees of "insiderness," and that the access control rules for an organization can be used to develop these degrees. We define someone as an insider *with respect to access to some data or resource* X.

3 Framework

Our framework for analyzing the insider problem consists of two components. The first, a model we call *Attribute-Based Group Access Control* (ABGAC) [4], captures the notion of what entities can access an object. The second is a model that describes policies at different layers of abstraction, much as operating systems have different layers of abstraction. Merging these two models allows us to characterize the insider. We begin with ABGAC.

ABGAC is a generalization of *role-based access control (RBAC)*. Whereas RBAC largely assigns rights based on specific job functions that a person has within an organization, ABGAC assigns rights based on general attributes, which might include elements of person's job function, but may also include a variety of other sources regarding the person or their environment.

We define attributes as descriptions of the protection domain of entities. The protection domain can include access rights to objects and resources such as systems, printers, documents, buildings, and generally any other object to which a user can have access. The protection domain can also include procedural access rights such as physical presence, or the ability to block access.

Once defined, the protection domains need to be partially ordered by value. The organization must do a cost/benefit analysis to assign a value to the

protection domains. For example, an organization might specify that access to financial documents, the email of senior level executives, and source code for specific products represents the information potentially of greatest value and, therefore, represents the greatest damage if leaked or compromised. The value of a protection domain of a user should not be defined solely by a systems administrator, but rather as a joint effort between the senior executives and the security administrators. Once ordered, the protection domains can be combined into groups, where the group indicates the threat level a particular set of attributes represents. These are called *pd-groups* to distinguish them from groups of users.

Paired with each protection domain is the group composed of the users to which that protection domain applies. In other words, groups are created based on the protection domains of the associated users, rather than on the job functions of the associated users (as in a role-based system). The users with access to the pd-groups with the highest value then represent those users who pose the greatest risk for insider threat. Given this pairing, we can create a lattice based on the ordering of protection domains and the ordering of groups. Given two pairs, we can determine which indicates the greatest risk by their ordering.

The creation of such a lattice requires a two-stage approach: determining the important components of the protection domain relevant to some privilege and identifying all users. It is not necessary to provide all components and privileges, but rather only those that are relevant to the well-being of the organization and therefore at risk due to insider threat. Initial users include not only direct employees, but also all contractors and out-sources (e.g., technical, clerical, janitorial), and any "special case" accesses (such as facility visitors or guest logins). Once the protection domains and users have been identified, the two are mapped together based on the access the users have. This creates an ordered group of users who represent insiders, where the ordering is based on the value of the resources to which they have access. Thus a security administrator can focus their attention on those insiders who pose the greatest threat, and an insider is thus defined with respect to the resources to which he has access.

Degree of insider abuse and job function may be very different, but we assert that the actual level of threat is much more important than the level of threat implied by a job function (which may or may not be true). By employing attributes and protection domains for the lattice rather than roles, we are able to specify and group disparate users who might have equal access in terms of insider abuse rather than simply by job function. For example, assume that the CEO of a company has identified the customer contact and purchase information as high-priority information that requires protection. Users who might have access to this information, and hence be placed in a group together, would include not only the sales representatives for the company, but also potentially external entities, such as the system administrators for SalesForce.com (assuming that the organization uses SalesForce.com). This is an example of the disappearance of a well-defined perimeter for an organization, and how ABGAC is able to still capture potential insider threats.

We have since extended the ABGAC model to include the notion of using policy discrepancies to identify insiders [3]. The extended model builds on Carlson's Unifying Policy Hierarchy [5], which defines four levels of policy (from highest to lowest): Oracle policies, Feasible Policies, Configured Policies, and Real-Time Policies, as shown in Fig. 1. The Oracle Policy is a policy that assumes perfect knowledge including the intent of a transaction. The Feasible Policy implements the Oracle Policy as best it can given real-world constraints (for example, a Feasible Policy will only be able to allow or disallow a transaction, but cannot determine the actual intent behind a transaction). The Configured Policy is the actual policy that has been implemented, since some Feasible Policies might be configured in multiple ways; the Configured Policy represents the choices made in implementing the Feasible Policy. Finally, the Real-Time Policy represents the policy decisions made on actual systems, and takes into consideration other issues such as system constraints or vulnerabilities. For example, if a buffer overflow elevates privileges, this is captured by the Real-Time Policy.

We argue that the ability to perform an insider at-

Unifying Policy Hierarchy		
Level	**Domain**	**Description**
Oracle Policy	all possible (s, o, a, e) tuples	Captures notion of an "ideal policy" even if such a policy isn't explicitly defined.
Feasible Policy	system-definable (s, o, a) tuples	Represents what can in practice can be captured on an actual system.
Configured Policy	system-defined (s, o, a) tuples	Represents the policy as configured on an actual system.
Real-Time Policy	system-defined (s, o, a) tuples	Represents what is possible on an actual system.

s: subject o: object a: action e: environment

Figure 1: Four levels of Carlson's Unifying Policy Hierarchy [5].

tack comes from a discrepancy in the expressiveness (and therefore enforceability) between two policy layers. As given in the example above, an Oracle Policy might specify the intent behind a transaction as determining if a transaction should be allowed, however the Feasible Policy has no ability to determine intent. Thus an insider might have the capability to perform a particular transaction, and might decide to abuse that capabiity as part of some insider attack. The discrepancy between the Feasible Policy (where the transaction is allowed) and the Oracle Policy (where the transaction is allowed only for specific reasons) allows this attack to occur. These two aspects of the model combine to provide a framework for discussing the insider threat problem, and for defining who is an insider.

From a practical, implementation perspective, we can merge the ABGAC model into a model of attacks for improved forensic analysis, simply by monitoring the use of credentials on sensitive documents. While security policies must identify sensitive documents and high levels of access to begin with, the forensic model helps to determine what is needed to understand the path to the objects, and the actions taken on them. The two models can also jointly identify where we cannot easily enforce policies by logging information, and thus provides a measure of how well an attempted threat to security can or cannot be determined in a post mortem analysis.

4 Case Study 1: Union Dime Embezzlement

In the years from 1970 to 1973, the Union Dime Savings Bank lost U.S. $1.5 million to embezzlement at the hands of Jérôme Kerviel, their chief teller [23]. The scheme would likely have lasted longer had there not been an unrelated arrest of Kerviel's gambling bookie, whose records resulted in his investigation and eventual conviction.

As chief teller, Kerviel was able to issue an "error correction" to accounts that reduced the digitally recorded account balance. He then pocketed the remainder in hard cash. When the time came for interest calculations, he would move money from other accounts into the account he 'corrected' so the balance would appear as expected and interest was properly calculated.

The embezzlement was enabled by several of the bank's practices. There were two types of accounts whose interest was calculated on different days, allowing money to be shifted from one account type to the other on the day interest was calculated. This allowed account records to appear balanced despite the teller's pocketing of money after issuing correc-

tions. In addition, customers received no monthly statements. An account's balance was recorded on a customer's booklet stamped at the time of deposit. Any adjustments to the bank's records would not be reflected until the customer's next withdraw, making low activity accounts an attractive target.

First, we fit this into our policy hierarchy. We emphasize that the following is one reasonable interpretation of the policy hierarchy. Others give similar results. We assume that the Oracle Policy states that "the chief teller can issue error corrections to accounts to correct errors in data entry." The Feasible Policy cannot distinguish between an "error in data entry" and "an error arising from illicit withdrawal." Thus, the Feasible Policy eliminates the motivation behind the error correction, and simply says "the chief teller can issue error corrections to accounts to correct errors." Here, the chief teller is complying with the Feasible Policy (because he is authorized to issue error corrections to accounts to correct errors) but not with the Oracle Policy (because the error being corrected is not related to an error in data entry; it arises from an illicit withdrawal).

Now consider a variant of the Feasible Policy that says "the chief teller can issue error corrections to accounts to correct errors, and shall record the reason for each error correction in a log." Now, when an auditor checks the accounts, the auditor can determine whether the chief teller issued the correction to fix a data entry error. But consider the next layer of policy, where the system is configured to record the log. If the log can only be made writable and not append-only, the chief teller can erase entries to hide that a change was made (and thus suppress the need to enter a reason). So, if the configuration policy says "the chief teller can write (edit) the log associated with error corrections to accounts," then there is a discrepancy between the Configured Policy (which says that the chief teller can change anything in the log) and the Feasible Policy (which says the chief teller's reason for changing the account must be recorded in the log).

Note also the discrepancy between the Configured Policy, the Feasible Policy, and the Oracle Policy. The Oracle Policy asserts that the chief teller's reason for changing the account is known, at least to the oracle, which can then decide whether the reason and the change comply with its policy. But the Feasible Policy says nothing about motive, merely that the teller record the reason for change. Similarly, the Configured Policy simply says the chief teller can write to the log, and nothing about *what* he must write. Hence there is a discrepancy on multiple levels: the chief teller can lie. Underlying this assertion is the Oracle Policy's ability to discern the actual reason for an act, and the inability of policies at other layers in the policy hierarchy to know the actual reason.

Given all this, we can integrate the ABGAC model to determine where insiders might arise. Let us assume the Oracle, Feasible, and Configured Policy as above. The resources involved in this episode of the Union Dime Bank are the cash in the bank, and the ability to take it physically from the bank; and the error correcting function and the ability to execute it. The tellers have access to the cash in the bank, as do those with access to the bank vault. Assuming the tellers are not searched when they leave, they also have the ability to take the cash physically from the bank. The question of who can execute the error correcting function limits the set of tellers to the chief teller, assuming correct implementation (a point we shall touch on in a moment). Thus, the set of people who can perform the above insider attack, namely embezzle funds in the manner described, is one: the chief teller.

The above analysis makes two assumptions. The first is that only one person is involved. The execution of the error correcting function requires the chief teller to act, so he must be involved in this compromise. But he need not be the one who takes the cash out of the bank. He could be in cahoots with one or more other tellers, who will remove the money for him. Such a compromise is possible, but less likely to succeed due to Benjamin Franklin's claim, "three may keep a secret, if two of them are dead" [10].

The second assumption is that the implementation of the Configured Policy is correct. For example, suppose there is a bug in the software managing the error correction routine. Then the Real-Time Policy is that anyone with access to the system on which that routine resides can change the amounts in accounts,

thereby performing the same function as the chief teller. Thus, the ABGAC analysis captures those attackers who exploit implementation bugs (or, more properly, discrepancies between the Real-Time Policy and the Configured Policy) in the same way it captures those who can exploit discrepancies between the higher layers of policy abstraction.

The application of ABGAC to this scenario provides a basis for identifying the threat. In doing so, it also provides a basis for mitigating the threat ahead of time, as well as instrumenting a system in a way that [18, 19] enables targeted logging of potential violations of the security policy, and therefore, more efficient analysis the whether a violation was attempted and successful. Specifically, the ABGAC model gives a benchmark of where logging is (a) feasible, and (b) useful. Where logging is not feasible, we can place bounds on the possible gaps in our levels of knowledge and attempt other forms of monitoring (e.g., physical security). Where logging is not useful, we can avoid taxing computer and network resources collecting useless data. In the place of the bank teller example, we can certainly isolate the bank teller and the systems and accounts that the bank teller has access to, and by generating attack graphs, starting with the teller's likely, ultimate goals (and/or the largest threats)—embezzlement—we can develop metrics that might help the other possible paths to accomplish those goals (e.g., collaborating with other bank employees), and monitor and protect accordingly.

Embezzlement is a particularly good demonstration of the insider model, and is broadly applicable in other such situations. For example, one might imagine that the French bank Société Générale wished they had been able to perform a better risk analysis by classifying insiders, threats, and targets using such a system before losing U.S. $7.1 billion [7].

5 Case Study 2: Social Engineering

Phishing as a security issue has traditionally been viewed as a social engineering attack and identity theft threat. However, it can also be viewed as a special case of the insider problem. In a phishing attack, the adversary sends an email to a target group soliciting them to perform some action that will reveal the target's credentials (to some target location) or sensitive information [1]. For example, the adversary might have set up a fake web site emulating a popular bank. He then sends email to some large number of email addresses, where the email appears to be an official communication from the bank. The email might encourage the user to "follow the link below" to log in and perform some action. As the link is actually to the fake web site, the adversary is then able to capture the credentials of the target user. Note that in this case the attack is indiscriminant and succeeds due to the large number of users targeted.

Related to phishing attacks are spear phishing and whaling [14]. Spear phishing refers to attacks that are targeted at particular individuals or companies, rather than indiscriminant as in generic phishing attacks. Whaling is a special case of spear phishing that is aimed at company executives.

In this section we examine two examples of phishing. First we consider the more generic (and prevalent) forms of phishing, and then we provide an example specific to spear phishing.

Given these descriptions, the insider attack occurs with respect to the target *organization* and not the individual. In the case of phishing, for example, the adversary might be trying to gain an individual's credentials in order to log into that individual's bank account and transfer funds to the adversary's PayPal account or make online purchases. Here the bank would be the target organization. The Oracle Policy in this case might be "Only the owner of an account can access and perform transactions using that account." In contrast, the Feasible Policy would state, "Only someone presenting the credentials of the owner of an account can access and perform transactions using that account." Note that the Feasible Policy cannot distinguish the owner of an account except through the use of his credentials. Thus any person providing those credentials is assumed to be the owner of the account. In this case the Configured Policy is the same as the Feasible Policy.

Translating this into the ABGAC model we have as resources the account at the bank and the money in that account, while the users of primary interest are the person owning that account, the employees of the bank, and the adversary.[1] In this case the account owner has access to the account and the money in that account, as do the bank employees, while the adversary does not. The insiders for this account are therefore the bank employees and the account owner. When the adversary obtains the account owner's credentials, he is the account owner from the perspective of the bank. Thus the adversary is also now an insider.

The examples for spear phishing (or whaling) are slightly different from the more general forms of phishing. In these cases the email exchange often aims to gain the trust of the target, enticing him to install some piece of software [11]. Phishing emails gain this trust by providing a sufficient amount of identifying information that the target believes the adversary is who he claims to be. The software installed is generally some form of malware, such as a keylogger, that sends information back to the adversary. This is interesting as an insider threat problem because, techically, the adversary *never* operates inside the perimeter of the target organization, however he does receive that organization's information (e.g., logged keystrokes, particular files). However, using the poicy discrepancy and ABGAC models, this case can be represented as an insider problem.

Assuming the case of a keylogger that sends information back to the adversary, the Oracle Policy might state "this computer can send information to other machines on the network upon explicit approval of the user." However, the Feasible Policy might be less restrictive: "This computer can send information to other machines on the network if the user is logged in." The assumption here is that if the user is logged in (which is easy to determine—at least that *some* user is logged in, using this user's credentials) then he approves of the communication being sent since he is (presumably) the one sending it. In contrast, requiring explicit approval for all communication requests would likely be onerous for the user (*e.g.*, needing to approve all web browsing activity, every email sent, etc.). Thus the computer program, which logs the keystrokes of the user and then sends the information back to the adversary, performs an insider activity. The adversary, although always external to the host machine and network, is the insider as he gains insider knowledge based on the keystroke logging he receives. This is made possible by the assumption that a user who is logged in is explicitly approving all communications between his computer and any outside systems.

This scenario can be translated into the ABGAC model by defining the resources as access to the particular system, the information on that system, and the new information entered and activities performed on that system. The users for this system are the user with legitimate access to this system (*e.g.*, the employee, CEO, etc.) and the adversary (along with others, such as system administrators). In this case, due to the presence of the keylogger, while the two users are separate with regards to access to the system and the information on that system, they can be grouped together regarding the new information gathered and activities performed on that system. Thus the adversary in this case is an insider.

6 Case Study 3: San Francisco's Password

A third example of the insider problem arises from acts of omission. In 2008, a San Francisco city computer engineer changed a password used for administrative access to a city network, and refused to reveal it [9]. The network, according to the City, handled payroll records and law enforcement records among other things. The inability to access the network severely impeded the operation of several critical governmental functions.

We now apply our framework. The Oracle policy states that "Authorized personnel shall be able to access the network to administer it". But in this

[1] Actually, others who would need to be considered include everyone who may have indirect access to the account, such as the account owner's spouse and other people living with the account holder. The specific details of how the account holder guarded his account would determine these.

case, the Configured Policy does not require that the password be set to something authorized personnel know; it merely requires that the user attempting to administer the network know the password in order to access those functions. In this case, the gap between *an authorized administrator* knowing the password (Oracle Policy) and the inability to express that as a configuration on the system (the Configured Policy) means that anyone who can reset the password is a potential insider.

The use of ABGAC shows how this works. The "resource" is the password. The "access" is the ability to reset it. The set of insiders therefore is the set of people who have access to the system, and authorization to reset the password; and anyone who can manipulate them to reset the password or give them access to do so (for example, perhaps by phishing). In this case, the culprit was identified as someone who did have direct access.

As an interesting note, the password was obtained by Gavin Newson, the mayor of San Francisco, who met with the engineer in jail and convinced him to reveal it [8]. Thus Newsom, had he access to the relevant network systems, would also be considered an insider under the ABGAC model. More interestingly, the district attorney made public about 150 user names and passwords used to connect to the City's network in a court filing [15]. The point here is that the model is powerful enough to encompass non-intuitive situations, such as the ability of the defenders to compromise the system. This reflects the old saying, "But who will guard the guards themselves?"

7 Related Work

The literature contains many frameworks for the insider problem. Some papers focus on the attacker. For example, Probst *et al.* [21] use a process algebra to model the actions of entities, and reason about them in the context of the insider problem. This work differs from ours in that they begin with an entity and then analyze what that entity can do, whereas our methods begin with the target and ask who can access it.

Chinchani *et al.* [6] use a graph-based approach, but focus on the target rather than the action. Their "key challenge graph" represents entities holding information or capabilities as nodes and channels of access or communication as vertices. It has an orientation similar to our approach, but the methodologies are completely different.

A second set of papers focus on characterizing insiders by characteristics of the entity involved. For example, Magklaras and Furnell [13] analyze the sophistication of end users to predict insider threats. This work is different than ours, although it could be applied to develop risk metrics that the ABGAC model can then use to quantify risk in certain situations.

A third set focuses on specific techniques for finding insiders. Examples include the use of honeypots [22] and anomaly-based intrusion detection mechanisms [20, 12, 17]. Our work is at a more abstract level, and can be used to guide the application of these technologies.

8 Conclusion

This paper has applied an approach for analyzing the insider threat to three simple situations. The framework discussed in section 3 applies equally well to either situation, and can in fact be extended to perform a risk analysis based on the probability of the individuals involved being an adversary, combined with the value of the effects they can cause due to their access. Unfortunately, determining these probabilities and values is difficult even when precise details of the situations suggest the analysis to determine them. As we simply wish to demonstrate the utility of the method in *describing* the insider, and show how the framework can be applied, we do not provide these probabilities or values.

In this paper we summarized our previous work and apply the approach to three case studies that demonstrate the utility of our approach. While our previous work has been largely theoretical, we focus here on the utility of our approach and its ability to represent insiders in different environments. We demonstrate how the model can be applied in both

traditional insider cases (e.g., embezzlement) as well as in social engineering threats (e.g., phishing). Old threats are increasing in severity due to the speed at which they can occur and the ability of computer networks to provide rapid, anonymous communication, whereas traditional threats required in-person, human contact, which was slower, more complicated, and more dangerous. In all of these case studies, we have shown the model allows for finer-grained and more appropriate classification of the threat scenarios. We also briefly demonstrated how the classification and identification of the threats can be merged with a model of attacks to guide a post mortem analysis.

Acknowledgements

Matt Bishop was supported by grant CND-0716827 from the National Science Foundation to the University of California at Davis. Sophie Engle was supported by grant H98230-07-1-0234 from the Department of Defense to the University of California at Davis. The views and conclusions expressed in this paper are those of the author, and not necessarily those of any funding agency.

Sean Peisert was supported by grant 2006-CS-001-000001 from the U. S. Department of Homeland Security, under the auspices of the Institute for Information Infrastructure Protection & I3P research program. The I3P is managed by Dartmouth College. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security, the I3P, or Dartmouth College.

References

[1] Binational Working Group on Cross-Border Mass Marketing Fraud. Report on phishing. Technical report, US Justice Department and the Ministry on Public Safety and Emergency Preparedness Canada, October 2006.

[2] M. Bishop. Position: "Insider" is Relative. In *Proceedings of the 2005 New Security Paradigms Workshop (NSPW)*, pages 77–78, Lake Arrowhead, CA, October 20–23, 2005.

[3] M. Bishop, S. Engle, S. Peisert, S. Whalen, and C. Gates. We Have Met the Enemy and He Is Us. In *Proceedings of the 2008 New Security Paradigms Workshop (NSPW)*, Lake Tahoe, CA, September 22–25, 2008.

[4] M. Bishop and C. Gates. Defining the Insider Threat. In *Proceedings of the 2008 Cyber Security and Information Infrastructure Research Workshop*, Oak Ridge, TN, 2008.

[5] A. Carlson. The Unifying Policy Hierarchy Model. Master's thesis, University of California at Davis, Dept. of Computer Science, 1 Shields Ave., Davis, CA, June 2006.

[6] R. Chinchani, A. Iyer, H. Ngo, and S. Upadhyaya. A target-centric formal model for insider threat and more. In *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, pages 108–117, 2005.

[7] N. Clark and D. Jolly. Fraud Costs Bank $7.1 Billion. *New York Times*, January 25, 2008.

[8] J. V. Derbeken. Lawyer says client was protecting city's code, July 2008.

[9] J. V. Derbeken. Not guilty plea from cybercoup suspect, 2008.

[10] B. Franklin. *Poor Richard's Almanack*. 1735.

[11] iDefence. Spear Phishing and Whaling Attacks Reach Record Levels. *iDefense Press Release*, June 2008.

[12] A. Liu, C. Martin, T. Hetherington, and S. Matzner. A comparison of system call feature representations for insider threat detection. In *Proceedings of the Sixth Annual IEEE Systems, Man and Cybernetics Information Assurance Workshop*, pages 340–347, June 2005.

[13] G. Magklaras and S. Furnell. A preliminary model of end user sophistication for insider threat prediction in it systems. *Computers and Security*, 24:371–380, 2005.

[14] J. Markoff. Larger Prey are Targets of Phishing. *New York Times*, April 16, 2008.

[15] R. McMillan. San francisco da discloses city's network passwords.

[16] A. P. Moore, D. M. Cappelli, and R. F. Trzeciak. The "Big Picture" of Insider IT Sabotage Across U.S. Critical Infrastructres. Technical Report CMU/SEI-2008-TR-009, CERT, May 2008.

[17] J. Park and J. Giordano. Role-based profile analysis for scalable and accurate insider-anomaly detection. In *Proceedings of the 25th IEEE International Performance, Computing, and Communications Conference*, Apr. 2006.

[18] S. Peisert, M. Bishop, and K. Marzullo. Toward Models for Forensic Analysis. In *Proceedings of the 2nd International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE)*, pages 3–15, Seattle, WA, April 2007.

[19] S. P. Peisert. *A Model of Forensic Analysis Using Goal-Oriented Logging*. PhD thesis, Department of Computer Science and Engineering, University of California, San Diego, March 2007.

[20] A. Phyo and S. Furnell. A conceptual framework for monitoring insider misuse. In *Proceedings of the Tenth Annual Scientific Conference on Web Technology, New Media Communications and Telematics Theory Methods, Tools and Applications*, pages 90–95, 2004.

[21] C. Probst, R. Hansen, and F. Nielson. Where can an insider attack? In *Proceedings of the Fourth International Workshop on Formal Aspects in Security and Trust*, Aug. 2006.

[22] L. Spitzner. Honeypots: Catching the insider threat. In *Proceedings of the 19th Annual Computer Security Applications Conference*, pages 170–179, 2003.

[23] T. Whiteside. *Computer Capers*. Mentor, 1978.