Advances in Information Security 49

Christian W. Probst
Jeffrey Hunker
Dieter Gollmann
Matt Bishop *Editors*

Insider Threats in Cyber Security

Springer

Insider Threats in Cyber Security

Advances in Information Security

Sushil Jajodia
Consulting Editor
Center for Secure Information Systems
George Mason University
Fairfax, VA 22030-4444
email: jajodia@gmu.edu

The goals of the Springer International Series on ADVANCES IN INFORMATION SECURITY are, one, to establish the state of the art of, and set the course for future research in information security and, two, to serve as a central reference source for advanced and timely topics in information security research and development. The scope of this series includes all aspects of computer and network security and related areas such as fault tolerance and software assurance.

ADVANCES IN INFORMATION SECURITY aims to publish thorough and cohesive overviews of specific topics in information security, as well as works that are larger in scope or that contain more detailed background information than can be accommodated in shorter survey articles. The series also serves as a forum for topics that may not have reached a level of maturity to warrant a comprehensive textbook treatment.

Researchers, as well as developers, are encouraged to contact Professor Sushil Jajodia with ideas for books under this series.

For a complete list of titles published in this series, go to www.springer.com/series/5576

Christian W. Probst • Jeffrey Hunker
Dieter Gollmann • Matt Bishop
Editors

Insider Threats in Cyber Security

Springer

Editors
Christian W. Probst
Technical University of Denmark
Informatics & Mathematical Modelling
Richard Petersens Plads
DK-2800 Kongens Lyngby
Denmark
probst@imm.dtu.dk

Jeffrey Hunker
5109 Bayard St
15232 Pittsburgh, PA
USA
hunker@jeffreyhunker.com

Dieter Gollmann
Technische Universität
Hamburg-Harburg
Institut für Sicherheit in verteilten
Anwendungen
Harburger Schlossstrasse 20
21079 Hamburg-Harburg
Germany
diego@tu-harburg.de

Matt Bishop
University of California, Davis
Department of Computer Science
Shields Ave. One
95616-8562 Davis California
USA
bishop@cs.ucdavis.edu

ISSN 1568-2633
ISBN 978-1-4419-7132-6 e-ISBN 978-1-4419-7133-3
DOI 10.1007/978-1-4419-7133-3
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010932010

Preface

Insider threats are easy to counter. One only needs a concise model of human behaviour and its dependencies on outer and inner influences, a surveillance system in place that is able to observe in necessary detail action and influences, and an evaluation system, that can draw the necessary conclusions from its input.

Neither of the components just described is easy to realise, or desirable to have in the first place. Modelling human behaviour is close to impossible, let alone modelling how it depends on outer and inner factors. A surveillance system is heavily dependent on legal boundaries of what is allowed to be monitored or not, and the amount of data even from legal monitoring can be overwhelming at best. An evaluation system would need to be able to take all the input and models into account, and this is yet another complex task.

This book collects a series of chapters that try to map the territory between modelling, analysing, and evaluating insider threat scenarios. The chapters cover aspects from insider threats in electronic voting, over monitoring and access control systems, to legal aspects and the integration of the approaches described into Information Security Management systems.

One important and recurring theme is the question of how much surveillance is admissible and acceptable in different settings. It is this question that in the end determines the success of techniques aiming to reduce insider threats, or threats in general. This is especially true when dealing with systems *beyond* the pure technical aspects, but towards psychological aspects.

We are indebted to the participants of the Dagstuhl Seminar "*Countering Insider Threats*" (08302), during which the idea for this book was first discussed, and the staff at Schloss Dagstuhl. It has been a long road, but we believe that the result, which you are reading now, was worth it.

Kongens Lyngby, Pittsburgh, Hamburg-Harburg, and Davis
January 2010

Christian W. Probst
Jeffrey Hunker
Dieter Gollmann
Matt Bishop

Contents

Aspects of Insider Threats

Christian W. Probst, Jeffrey Hunker, Dieter Gollmann, and Matt Bishop

Abstract The insider threat has received considerable attention, and is often cited as the most serious security problem. It is also considered the most difficult problem to deal with, because an "insider" has information and capabilities not known to external attackers. The difficulty in handling the insider threat is reasonable under those circumstances; if one cannot define a problem precisely, how can one approach a solution, let alone know when the problem is solved? This chapter presents some aspects of insider threats, collected at an inter-disciplinary workshop in 2008.

1 Introduction

The "insider threat" or "insider problem" has received considerable attention [2, 13], and is cited as the most serious security problem in many studies. It is also considered the most difficult problem to deal with, because an "insider" has information and capabilities not known to other, external attackers. However, the term "insider threat" is usually either not defined at all, or defined nebulously.

The difficulty in handling the insider threat is reasonable under those circumstances; if one cannot define a problem precisely, how can one approach a solution, let alone know when the problem is solved? It is noteworthy that, despite this imponderability, definitions of the insider threat still have some common elements. For

Christian W. Probst
Technical University of Denmark, e-mail: probst@imm.dtu.dk

Jeffrey Hunker
Jeffrey Hunker Associates, e-mail: hunker@jeffreyhunker.com

Dieter Gollmann
Hamburg University of Technology, e-mail: diego@tu-harburg.de

Matt Bishop
University of California, Davis, e-mail: bishop@cs.ucdavis.edu

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_1,

example, a workshop report [4] defined the problem as malevolent (or possibly inadvertent) actions by an already trusted person with access to sensitive information and information systems. Elsewhere, that same report defined an insider as someone with access, privilege, or knowledge of information systems and services. Another report [12] implicitly defined an insider as anyone operating inside the security perimeter—while already the assumption of only having a single security perimeter may be optimistic.

In 2008, a Dagstuhl seminar on insider threats brought together researchers and practitioners from different communities to discuss in a multi-national setting what the problems are we care about, what our response is, which factors influence the cost of dealing with insider threats and attacks, and so on. In a time where we barely understand which factors cause insider threats, and our solutions are scattered all over communities, areas, and instruments, this coordinated action between the involved communities seems to be needed more than ever.

This chapter presents some of the results of that workshop, where also the idea for this book was born. Many of the aspects identified in this introductory chapter are touched upon throughout the book. An earlier version of this chapter, as well as more information on that seminar, is available from [9].

2 Insiders and Insider Threats

One of the most urgent quests for communities dealing with insider threats is identifying the characteristic features of an insider. One approach for doing so is to look at recent insider threat cases, and try to find individual or common properties. This is an important step, since insider threat cases can be rather diverging—take for example cases such as Binney vs. Banner [1], a message flood created as consequence of a security bulletin [11], spies that stole secrets for the Chinese Army [7], or a tax authority employee who used her influence to embed backdoors into taxation software [10] (see boxes below for short summaries). While these cases could not differ more, they serve the purpose of illustrating the widely differing characteristics of insider threats.

The wide range of properties that can characterize insider threats recently has led to the development of taxonomies, which are discussed in Section 2.2. Especially Case 2 (message flood) is interesting, as it seems unclear whether this really is an insider case, and if yes, whether it was the deed of a single insider, or a confluence of several actions by insiders. Case 4 (taxation software), on the other hand, seems typical for an employee who is an insider, but needs to "break into" the system to reach certain goals.

To be able to deal with cases so divergent, one clearly needs 1) a common vision of how insiders can be categorized; and 2) security policies for countering insider threats, and ways to evaluate the impact of alternative security policies.

From analyzing cases such as the above, several approaches to identifying an insider can be developed:

Example 1: The Hard Disk Example: Naive user and absent policy
On April 5, 2003, Banner Therapy (a small privately owned company in North Carolina, USA) employee Christina Binney was discharged from her position for "misconduct", and instructed not to return to the office.
Christina Binney was also a co-founder of Banner Therapy. According to Banner, there were two reasons for Binney's dismissal. First, the company disputed her assertion of copyright interest in the company catalogue and website. Second, the company claimed she impermissibly removed from her work computer a hard drive that she took home over the weekend to prepare for a client meeting. The company claimed that the disk drive removal crippled Banner's operations and placed vital company data at risk. Binney explained that a Banner customer requested a meeting on a Friday for the following Monday morning. To prepare for the Monday meeting, Binney chose to physically remove the entire hard drive from her work computer to use with her compatible home computer, rather than take time to transfer the files to a disk.
At the time, Banner Therapy had neither company policy about taking work equipment home nor established computing protocols. When Binney attempted to return to work on Monday, she was denied access; this inability to enter the workplace prevented her from returning the hard drive as she claimed she intended to do.

Example 2: The Email Example: Ordinary user generates an extraordinary amount of email
In early October 2007, Alex Greene was changing jobs. In preparation for the switch, he wanted to update his subscription to a Department of Homeland Security intelligence bulletin by changing his designated email address. In doing so, he mistakenly hit "reply all", and touched off a listserv free-for-all when his request arrived in the electronic mailboxes of several thousand government and private sector security specialists. The result was what commentators described as a mini-distributed denial of service attack. There were more than 2.2 million emails pinging among approximately 7,500 recipients before the email server was forced to shut down.
The information contained in the bulletin is unclassified, but nevertheless, the decision to respond inadvertently compromised classified contact and departmental information. Individual subscribers with security classifications remained anonymous until they also hit reply, responding from work accounts that included automatically generated signatures. Indeed, one poster pointed out that, armed with the information contained in auto-signatures, he was one fake letterhead away from impersonating a Department of Defense employee.

Example 3: The Trade Secret Example: Malicious user steals trade secrets
On June 16, 2007, FBI agents, using a sealed grand jury indictment, entered two luxury homes in Silicon Valley and arrested a pair of engineers. Both Lan Lee (an American citizen) and Yuefei Ge (a Chinese national) had worked for NetLogic Microsystems (NLM) until July 2003. The two men used money from mainland China to create and incorporate a company for the sole purpose of exploiting the secrets they stole.
Lee and Ge downloaded sensitive NLM documents onto their home computers. NLM data sheets are "top-level confidential technical descriptions of their products", including information described in enough specificity to enable someone to produce the technology. Together, the men accumulated the information needed to design and produce their own lines of microprocessors and microchips. To finance the business they were creating, the men contacted Beijing FBNI Electronic Technology Development Company Ltd, and entered into an agreement to develop and sell microprocessor chips. Both men were able to access proprietary information without exceeding their individual authorizations.
By late September investigators had uncovered evidence that the venture capitalist had ties to the Chinese government and military.

Example 4: The Tax Fraud Example: Perimeter definition and system design
The District of Columbia (as of summer 2008) is pursuing a case against Harriette Walters and her co-conspirators, for perpetrating the biggest fraud in the city's history. Until her arrest, "Walters was a 26-year tax employee known among her colleagues as a problem solver with a knack for finding solutions by using the department's antiquated and balky computers or finding a way around them." She allegedly used her position to produce fake checks for bogus refunds with fictitious names; the total is said to exceed (USD) $50 million.
The scheme involved Washington's new Integrated Tax System. During design phase, Walters "contributed to the decision that her unit, which handled real estate tax refunds, be left out of it." At the time, the decision seemed to make sense. D.C. had spent $100 million to implement the business and income parts of the system, and it had only $5 million remaining for implementing the real estate tax portion. So the system's perimeter was defined to omit real estate tax processing. That design decision allowed Walters and her co-conspirators to create bogus tax refunds with fictitious names that were not checked against actual real estate records. Some refunds were issued multiple times; the recipient (often someone's boyfriend) would claim that the check was never received, and a new one was issued—with interest to compensate for the long delay! The schemes exploited several loopholes: each check was under the $40,000 threshold for requiring a supervisor's approval, and no action was taken to cancel the first check or confirm that it had not already been cashed.

- An insider is defined with respect to a resource, leading to "degrees of insiderness";
- An insider is somebody with legitimate access to resources;
- An insider is a wholly or partially trusted subject;
- An insider is an individual who has or had access to resources;
- An insider is a system user who can misuse privileges;
- An insider is an individual with authorized access who might attempt unauthorized removal or sabotage of critical assets or who could aid outsiders in doing so; and
- An insider is a person or company whom we trust.

These definitions immediately lead to a series of discussions on what is meant by "access" (code, credentials, timing of access rights), whether an insider is sufficiently defined based on resources or whether a definition should take the system into account, and how the definition relates to a masquerader, namely an outsider being able to trick a system into believing he is an insider.

Exploring these aspects enables us to reason about what makes a good insider:

- Knowledge, intent, motivation;
- Possesses power to act as agent of the business;
- Knowledge of underlying business IT platforms;
- Knowledge/control over IT security controls; and
- Ability to incur liability in pecuniary terms or in brand damage or other intangible terms.

The skill of insiders is also an important a factor defining the threat posed by malicious insiders, or non-malicious insiders just trying to get their job done. "Motivation" in general is an important question when dealing with insider threats and their consequences. This can cover the whole range from "innocent action", "fun",

"technical challenge", "criminal intentions", to "espionage", or a combination of each of these factors. Surprisingly, even though one would expect the contrary, the effect of actions can be equally devastating for each of these motivations. This, of course, makes detecting a threat even more important—but also more complicated. A key observation is that the definition of an insider for threat purposes is different than the definition for business purposes.

Based on the aspects defined above, one can in turn decide how to defined an insider, namely in terms of someone with:

- Knowledge: Implies an open system, one that remains secure (if at all) even with full knowledge of the system operation; alternatively, security through obscurity; or
- Trust: An individual is empowered by the organization to be an insider; or
- Access: An insider is in possession of a credential giving access to the system — an IT centric perspective, since the system in general does not know who possesses the credential.

At the end of the Dagstuhl seminar [9], a trust-based definition of an insider was proposed:

> "An insider is a person that has been legitimately empowered with the right to access, represent, or decide about one or more assets of the organization's structure."

The rationale behind this definition is that it removes any specific IT bias from the definition, it focuses on organizational assets rather than a narrow approach based on system credentials, and while people that constitute threats may not be entrusted access to credentials, they might still have the ability to decide (based on policies) and to represent the organization.

The ability to represent is rather important, as the policies imposed on an actor inside an organization in general are not known to the outside, where the same actor can pretend to be subject to completely different policies—a factor rarely ever being checked.

"Knowledge" by an individual (*e.g.*, the knowledge of the person who originally designed the system, but is not part of the organization or in any way associated with the organization anymore) is not a good way of capturing what is an insider.

2.1 Insider Threats

A natural consequence of having defined the term "insider" is to consider the term "insider threat". As discussed in the previous section, again many different aspects can be important:

- Risk to organization or organizational resources posed by actions (the entity as agent of the business);
- Access type or system role;
- Aim or intentionality or reason for misuse;

- Level of technical expertise;
- Type of malicious insider behaviour or system consequences; and
- Threats from masquerader, traitors, and naive insiders.

The last point of masqueraders (individuals pretending to be legitimate insiders but without valid access), traitors (legitimate insiders acting in malicious ways), and naïve insiders (who cause damage without malicious intent) is very closely related to the question of motivation discussed above. The problem of interest is dealing with the "real real insider"—an individual deeply embedded in an organization (*e.g.*, a high level executive, or a systems administrator). Detection techniques for each of these types of insider threats will vary.

The impact of insider threats can occur in multiple dimensions: financial loss, disruption to the organization, loss of reputation, and long term impacts on organizational culture. These impacts can be highly nuanced, and are neither well or easily measured or accounted for. An important aspect here is that "bonus round" insider threats (actions taken in anger, revenge, spite regarding bonuses, compensation) can have severe consequences on all levels of an organization. Thus a rather "small" or meaningless motivation for the individual can have a rather huge impact for the organization. Equally, the impact may not depend on motivation—an innocent act can have as devastating an effect as a maliciously motivated attack. The goal of detecting insider threats may therefore be to avoid catastrophic consequences regardless of the motivation. These aspects as well as other risk accelerants should be represented in threat models, to acknowledge their importance.

2.2 Taxonomies

In order to allow identification of insiders and insider threats across organizational boundaries, an effective taxonomy of insider threats is a necessary foundation for further work by both researchers and practitioners. Taxonomies provide a means to order problems; in the problem considered here such ordering is necessary both to differentiate types of insiders and types of insider threats, and to make explicit the key dimensions which serve as the basis of the differentiation. By identifying the key dimensions, we can then begin to systematically build prevention and response strategies.

At least some common acceptance of the dimensions of such a taxonomy is needed; experts can disagree about how the dimensions are applied (as in defining who is an insider), but by forcing an explicit discussion of different interpretations, taxonomies can serve a vital role.

Examples for taxonomies specifically aimed at insiders can be found in [14] and [3]. Pred *et al.* [14] observe that insider threats can be defined in terms of four reference perspectives, namely the organization, the individual, the information system, and the environment. This results in a "holistic" or top-down taxonomy. In contrast, Bishop *et al.* [3] define insiders with respect to a resource. Resources are defined as pairs of the resource itself and access privileges, and insiders are defined

with respect to these pairs. With this structuring, it is possible to define degrees of insiderness.

An important question when considering taxonomies is whether a single taxonomy should be adopted by the community. In some cases, like the taxonomy for speciation (kingdom, phylum, ...) a more or less universal adoption has provided great value, but it can also limit a frameworks expressiveness.

Key factors important as determinants of the insider threat may be difficult to categorize a priori. As noted elsewhere, knowledge of the insider's intent is desirable, but requires all-embracing knowledge. Each determining factor for an insider can be used for defining a taxonomy, for example based on:

- The distinction between malicious and accidental threats;
- The distinction between doing something intentionally (for malice, or good reasons which nonetheless may result in damage) versus events that occur accidentally.
- The distinction between obvious and stealthy acts.
- Acts by masqueraders, traitors and naïve or accidental use that results in harm.
- A combination of factors such as access type; aim or intentionality or reason for misuse; level of technical expertise; and the system consequences of insider threats.

3 Detection and Mitigation

Forensics appears to be highly undeveloped when addressing insider threats. Insider behaviour may be close to the expected behaviour, and the still often used audit trail is generally inadequate (redundant, misleading, missing data), and often lacks time correlation. The number of appropriate characteristics to observe may be large, resulting in overwhelming amounts of data. While we have decent tools as the result of a large body of work on intrusion detection, it is unclear how these tools help with insider threats. Current forensics tools often require assumptions such as "only one person had access" or "the owner of the machine is in complete control". Therefore, forensics remains an art, and as an art questions such as what to log or determining the relevance of log data elude clear answers.

Detection, forensics, and response must also wrestle with how to distinguish motive and intent. Malicious acts may be equivalent to acts due to accidents or naïveté. Insiders may legitimately use domains in unexpected ways that might trigger false alarms. Outsiders, insiders acting with malicious intent, insiders acting without malicious intent, and accidental behaviour may all result in similar effects on the organization. Hence there are always going to be gray areas in how security policies define both insider misuse and proper behaviour. Furthermore, actions are context bound, but most security polices only inadequately capture the nuances of context.

Monitoring. While monitoring can help with technical aspects, it does potentially worsen behavioural aspects. The deciding factor is how much monitoring is acceptable (both ethically and legally), and whether it is at all beneficial. The noteworthy

point here is that this question arises at all levels in an organization, from individual actors, to groups, to companies, to the society as a whole. The problem is that not only may the same actor have different opinions depending on at which level he is asked, but also that different answers for different individuals may exist at the same level.

An interesting observation is that in certain settings with significantly enhanced monitoring, the number of identified incidents has stayed almost constant. At the same time, and even more worrying, cases such as Kerviel and the Liechtenstein case [8, 15] had in common that the attacker intimately knew the monitoring system and knew how to play it. It is often hypothesized that malicious insiders seek to avoid setting off monitoring alarms by slowly adjusting their profiles, but it seems unclear how easy current behavioural systems can be tricked.

In summary, trust in insiders is a behavioural expectation that still needs to be controlled. While the easy solution to reducing the number of insider cases would be to remove all restrictions (making the illegal actions legal by changing the semantics of the term "legal"), we aim for making the monitoring as efficient as possible, where in different situations the term "efficient" may have different interpretations. An important aspect that can not be underestimated are legal restrictions and privacy aspects of data collection, which may be even harder to follow in multi-national settings.

The goal of monitoring (or observing in general) should be to only monitor what is needed to identify the threat in question. Since currently trust can often be transferred, for example by handing over a code card, it is important to isolate transferred trust as much as possible, not least to allow the result of monitoring to be used to bind actions to actors.

In large complex systems risk analysis and focused detection require a significant effort. Not only may monitoring affect trust within an organization, it also is highly nuanced what to look for [16]. For example, an inordinate amount of searching may indicate a masquerade attack (the masquerader is less familiar than the legitimate insider about data structures)—or a forgetful mind. Observations on "higher levels of behaviour" not observed by systems monitoring may be useful, for example, by using human intelligence to pick up novel attacks and signals that are out of the system. CERT data suggests [5] that in most cases someone else knew about the insider threat actions going on, so it is important to find ways to encourage reporting of "suspicious activity" by others. Looking for suspicious (different) behavioural patterns by insiders is appealing, but difficult to systematically apply; behavioural patterns include cyber activity, physical movements, physiological signals, and many more. Employment screening data and self/organizational reported data might be useful here, but any screening for behavioural changes is bound to produce false positives from otherwise innocent factors like individual predispositions or lifestyle changes. Fundamentally, the attributes key to insider threat identification will be largely context bound.

From a company point-of-view it turns out to be often preferable to not mitigate ongoing insider attacks for numerous reasons. Here optimistic access control is seen as a viable option, *i.e.*, allowing insider actions to happen until there is no way back,

or even letting them happen unhindered, at the same time ensuring that enough evidence is collected through monitoring. It seems often more ruinous to take systems down because of an ongoing attack than to accept the losses and prosecute after the fact.

Outsourcing. Even more difficult is the handling of outsourced parts of a company, both for technical and legal reasons. On the one hand data may be much harder to obtain (or be less trustworthy), at the other hand the data protection laws regulating data collection in other countries may be vastly different from the laws in the home country.

Another problematic area is outsourcing the auditing itself; while in the "regular" outsourcing scenario it may be difficult to obtain the data in the first place, it now is paramount to protect the already collected data. This means the data should be anonymized as much as possible, revealing only as much data as necessary to allow external auditors to produce meaningful results, but at the same time hindering them from drawing unwanted inferences from the data. One example is to anonymize timestamps to preserve relative ordering between events, but blurring them such that the exact order and timing is lost. It was noted that formal methods can and should be applied in these settings, and some were presented, but at the same time often require significant resources, such that for example before and after the fact application is often feasible, but not online detection.

4 Policies

Policies obviously play an important role with respect to insider threats, as they define the boundaries between permissible and not permissible behaviour, both on a technical and non-technical level, and tie together insiders, insider threats, and detection and mitigation.

Policies not only define proper behaviour, but implicitly also define the notion of insider. It is problematic that policies often are only specified implicitly, possibly leading to large differences between *de facto* policies and "real", intended policies. To support the externalization of these intended policies, policy languages have been developed, which are usually quite well suited to support technical issues, and at the same time try to add support for non-technical aspects of policies.

When considering policies, one needs to pay special attention to the notions of "context" and "dynamicity". For example, a given actor might be an insider in one situation, but would be considered an outsider in another. Similarly, for some policies violations in special, emergency cases might be acceptable, but in the general case they should be observed. In this case it would be the insider's margin of discretion to decide for or against breaking a policy rule. This ties policies as well as their specification and enforcement tightly to human factors, which are discussed in the next section.

Policy Hierarchy. Policies themselves are developed based on three sources: 1) legal and regulatory (so-called best practices); 2) business requirements; 3) security

requirements. All of these sources can result in implicit or explicit policies, establishing a grey zone where behaviour is neither good nor bad. This potential gap is extended and formalized in the Unifying Policy Hierarchy [6], which established four different levels:

- Oracle Policy. Given perfect knowledge, what would policy be? Deals with inherent vulnerabilities.
- Feasible Policy. With imperfect knowledge, implement oracle as good as possible. Deals with configuration vulnerabilities.
- Configured Policy. What the system implements via configuration. Deals with real time vulnerabilities.
- Real Time Policy. Add in security vulnerabilities.

Gaps and conflicts in policies can be considered as a principle factor in allowing insider threats to occur; in some cases because gaps/conflicts create confusion among insiders in terms of "what is right" or "how do I get my job done"; in other instances because gaps/conflicts create opportunities that malicious insiders can exploit.

To acknowledge the risk of gaps between policies, we need an analysis of specifications for gaps and conflicts. Reasoning about insider threats it becomes apparent that policies normally do not make explicit who an insider is—an obvious requirement if we want to be able to analyse their hierarchies and fine-tune their impact. If we have policy-language support for specifying the roles of actors, then one may classify certain requests as coming from insiders, or in general build in context-dependent handling of insiders versus outsiders. For example one might want to be able to express that certain requests may only come from insiders, or on an even more context-dependent level, what degree of insiderness is required for a certain behaviour to be permissive.

Policy languages. A gap of a different kind exists for policy languages. Here the gap exists between the existing capabilities to specify system (and more broadly, organizational) policies, and the needed qualities of policy to adequately prevent insider threats. One of the most urgent needs, already mentioned above, is for policies to be aware of behavioural aspects and context, thereby being able to handle and regulate abstract events. While a "zoo" of policy languages exists, with a vast overlap in terms of what they can achieve, the user often may not be able to write policies, let alone read, understand, and follow them. This is expected to gain growing importance in future interactions between previously independent parties.

As mentioned above many systems define the notion of insider relatively to system boundaries. In the long run we may therefore need domain-specific policy languages, in which for example actions would be allowed only if discretionary circumstances justify their execution.

From a research perspective we often seem to be unaware of the different levels that the same policy may exist in, but instead take for granted that an oracle security policy is provided; given "the" security policy, it is often assumed that this resolves all tensions between organizational culture, work flow, and compliance by (implicitly) enforcing for compliance with security practices for the sake of security.

However, having some policy is not enough, that is deploying security technology and policies does not automatically help in achieving security. This is especially true due to the above-mentioned context-dependency of policy rules.

5 Human Factors and Compliance

When considering human factors and compliance, it seems that most insider threat policies are based on a set of incorrect assumptions, namely, that 1) once someone is vetted, audit and incident management processes will pick up policy violations; 2) that risk assessment processes will pick up changes to individual and group values; and 3) that training and awareness-building education programs will instill desired security culture. However these assumptions never fit to how people actually want to pick up information, or act in the course of doing their jobs within the organization. The inadequacy of many existing security solutions to address real life human behaviour presents us with a set of challenges on how to better incorporate human factors into solutions.

Second, an important problem is to align security policies with organization workflow, or, stated simply, security should support people doing their jobs. Often technological security approaches are not accepted and in fact actively subverted, because they interfere with work flow (*e.g.*, an iris reader with an "unacceptable" delay before allowing access resulted in staff finding other ways of gaining access). Compliance with security policies is hard; to make compliance easy for insiders is absolutely necessary for any successful effort to constrain insider threats. Compliance (defined as efforts users will make for purposes they do not understand or agree with) is limited; getting compliance gets more expensive the closer you get to the limit of peoples' tolerance for disruptions to their work flow and social interactions. Successful security policies need to demonstrate to insiders the value of security, not just the requirement for security.

Another important observation is that motive and intent matter a great deal, but multiple motivations may map into a single intent. As a simple example consider the act (intent) to prop a door open. The motive for this action might be benign (being lazy, or carrying large packages into the room) or malicious (propping the door open to allow unauthorized persons to enter). Observables may be able to capture the intent but not the motivation. This has important implications on the limitations of monitoring, and highlights again the need to establish context for specific actions.

Understanding and Integrating Human Factors. Criminology can inform insider threat understanding, and within criminology are several theories relevant to insider threats. Earlier theories of deterrence, social bonds, and social learning have been integrated into a theory of planned behaviour: for a crime to be committed a person must have both motive and opportunity. As just noted, motive matters; for the "cold intellectual attacker" when the possibility of punishment is high and the sanction severe potential criminals will be deterred from committing illegal acts, especially when their motives are weak. More generally, the goal of "situational" crime pre-

vention is to 1) make the criminal act appear more difficult; 2) make the criminal act more dangerous; 3) reduce the benefit a potential criminal is expecting to recover; and 4) remove the excuses available to the potential malefactor.

Organizational purpose and management structures affect both security structure and policy. In discussing organizational factors relevant to the insider threat a number of questions must be considered:

- How does trust grow in organizations? In some organizations for example there is lots of trust at the base of the organization but it does not necessarily rise up.
- How can organizations adjust management processes to engender a more positive environment for security? Specifically, how can organizations develop a "reflexive view" that looks at the whole person rather than just as a work resource?
- Whistleblowing: When are organization members comfortable with whistle blowing? Is there are role for technology in extending the whistle blowing capabilities of staff?
- Policy conflict within organizations: It seems reasonable to assume that all organizations have implicit tradeoffs about what is more and less important in their expressions of policy. How can these be made more explicit so that policy and security architectures can more effectively capture these values? Doing so might require a hierarchy of organizational needs like the Maslow hierarchy of individual needs.
- Organizational clustering: how much do organizational units cluster in their values? Are there psychological contracts by group clusters within organizations that can be mapped by looking at risk behaviours?
- How can we build robust policy so that when conflicts do arise they can be resolved efficiently in the best interests of the organization?

Insiders (or people in general) will act unexpectedly. Thus, flagging potential insider threats based on departures from "normal" patterns may lack reliability; monitoring for "out of normal" actions may generate too many false positives. There will also always be "gray areas" in drawing the line between insider misuse and proper behaviour.

Hence, context of an activity matters a great deal in accurately characterizing abusive insider actions. Context is defined in terms of physical, social, cultural, and temporal dimensions. In adding context into the shaping of security policies, the implication is that there are no standard solutions for workable security — what is usable is what fits. We need security policies appropriate for a specific domain (context) but what happens when insiders use domains in unexpected ways? Security controls must be characterized in the context of the activity, and because there will always be gray areas, those defining security controls must resist the temptation to believe that controls can eradicate all risk. Those defining controls must do this with full participation of management. Those enforcing controls must be willing to accommodate managerial discretion in certain settings.

The unpredictability of human behaviour has its implications for the role of trust in an organization. As stated earlier, trust is a behavioural expectation, and trust is only necessary in an organization when behaviours cannot be controlled in all

dimensions. Trust is also transitive, so one could argue that reducing insider threats would require environments where no one is trusted, or at worst only a few people are trusted; in any event transferred trust relationships should be eliminated.

Policies and Human Factors. Policies need to be shaped and evaluated in terms of their human impact. How specific should policies be? There is a perception that there are too many policies. The psychological contract with employees generally means that 1) policies need to be made more manageable, and 2) that there is a need to find a way of testing policies to remove redundant policies. The ideal would be a small set of consistent security policies related to behaviours, and fit with business processes, and organizational values and norms.

A common theme is the need to link the user community (the insiders in the organization) with the policies being developed and enforced. Failing to engage staff in security may be the norm, but this lack of engagement weakens security. Security will only work in organizations where people feel that they are part of a larger community. Organizations could conduct specialized internal exercises with most or all the insiders to identify both the set of useful and acceptable policies, and unique contexts which may result in generalized policies in conflict with organizational needs. Equally it will be key to monitor the implementation of policies "on the ground" by engaging staff and managers on whether policies are appropriate, or interfere with their workflow or culture. Sustained discourse with insiders can help highlight positive examples (of senior executives, for example) and in myth busting; an important goal here is to remove frequently made excuses.

Issues of what we can measure, what is relevant to measure, and how and when we intervene when suspecting threatening insider actions need to take human factors into account. Consider the impact of false accusations of insider threats on both the individual and the organization. Many suspicious activities which can be observed are correlated with insider threat behaviour, but not causally linked. False accusations have multiple deleterious effects: investigative resources are spent, the individuals so accused may quit, seek legal or other recourse (including becoming a real insider threat!), or be affected psychologically, the organization's culture may be affected, possibly for extended periods. There is, therefore, a need for decision processes to decide when to intervene and how.

6 Conclusion

The Dagstuhl seminar on Countering Insider Threats improved our understanding of what different communities mean by "insider". As stated above, this knowledge has already during the seminar been used to develop integrated approaches towards qualitative reasoning about threats and possible attacks. Beyond this shared definition of what constitutes an insider, the most prominent outcome of the seminar is the beginning of a taxonomy or framework for categorising different threats. The seminar identified the need for:

- A framework or taxonomy for distinguishing among different types of insider threats;
- Methodologies for assessing the risk and impact of insider threat incidents;
- Incorporating human factors into the development of solutions;
- Better formulations for specifying useful policy at both systems and organizational levels—policy that would be meaningful and applicable to the insider threats deemed most important.

There were some cross-cutting conclusions that emerged from the seminar. The role of trust was discussed in a number of different contexts. In one sense, the ideal security framework for addressing insider threats would eliminate the need for trust—all behaviours would either be defined permissible, or else made impossible to execute. But this model ignores two realities. In any but the simplest settings, context of actions is highly determinative in shaping what is appropriate or needed behaviour. Further, many (most?) organizations would not accept a working environment so rigidly defined as to eliminate the need for trust. Hence, we emerge with the conclusion that trust relationships will be present in most organizations; how to best factor trust into security policies and frameworks remains, however, unclear.

Security, moreover, is context dependent. Security is not achieved by deploying generic (context free) controls. However, the importance of context in addressing insider threats poses a number of challenges. Capturing qualitatively the various situations that might arise in an organization is itself probably impossible, though effective dialogue between those defining security controls and those working as insiders in the organization will certainly help. Hence, insider threat prevention and response has to deal with the reality that controls will not adequately capture all of the behaviours that might be appropriate in a given context. Even if all contexts could be qualitatively described, policy languages and controls are inadequate at the current time to fully capture the range of contexts identified.

Motivation and intent clearly are important in defining insider threats and defining appropriate detection, forensics, and mitigation strategies. While intent (the purpose of actions) is at least partially observable, motivation (the incitement to action) is not. The intent to, for instance, obtain certain data may reflect malicious motives, or may reflect positive motives (as in a hospital emergency where certain information is desperately needed regardless of legitimate access). Devining motivation highlights the need for context-aware policies, but even with context motivations may be difficult to determine. We conclude that approaches for understanding motivation a priori are still highly immature.

Each of these observations emphasizes the conclusion that security will not be achieved solely by deploying security technology. Most people are not entirely logical or consistent in their behaviour, and this confounds our ability to formulate measures to reliably prevent or detect malicious insider behaviour.

References

1. Binney v. Banner Therapy Products, 631 S.E. 2d 848, 850. North Carolina Court of Appeals (2006)
2. Bishop, M.: The Insider Problem Revisited. In: Proceedings of the New Security Paradigms Workshop 2005. ACM Press, Lake Arrowhead, CA, USA (2005)
3. Bishop, M., Engle, S., Peisert, S., Whalen, T., Gates, C.: Case studies of an insider framework. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS) (2009)
4. Brackney, R.C., Anderson, R.H.: Understanding the insider threat : proceedings of a March 2004 workshop. RAND, Santa Monica, CA : (2004)
5. Cappelli, D.M., Moore, A.P., Shaw, E.D.: A Risk Mitigation Model: Lessons Learned From Actual Insider Sabotage. In: Computer Security Institute, 33rd Annual Computer Security Conference and Exhibition (2006)
6. Carlson, A.: The unifying policy hierarchy model. Master's thesis, Department of Computer Science, University of California, Davis (2006)
7. Cha, A.E.: Even spies embrace china's free market. Washington Post, February 15, 2008. Available from http://www.washingtonpost.com/wp-dyn/content/article/2008/02/14/AR2008021403550.html, last visited March 2010.
8. Hawley, C.: The liechtenstein connection—massive tax evasion scandal in germany. Spiegel Online International, 18 February 2008. Available from http://www.spiegel.de/international/business/0,1518,535768,00.html, last visited March 13, 2009.
9. Homepage of Dagstuhl Seminar 08302: "Countering Insider Threats". Available from http://www.dagstuhl.de/08302, last visited December 4, 2008 (2008)
10. Keating, D.: Tax suspects guidance on software left d.c. at risk. Washington Post (2008)
11. Kirk, J.: Homeland security e-mail server turns into spam cannon. InfoWorld.com, October 4, 2007. Available from http://www.infoworld.com/d/security-central/homeland-security-e-mail-server-turns-spam-cannon-924, last visited March 2010.
12. Patzakis, J.: New incident response best practices: Patch and proceed is no longer acceptable incident response procedure. White Paper, Guidance Software, Pasadena, CA (2003)
13. Pfleeger, S.L., Stolfo, S.J.: Addressing the insider threat. IEEE Security and Privacy **7**, 10–13 (2009). DOI http://doi.ieeecomputersociety.org/10.1109/MSP.2009.146
14. Predd, J., Pfleeger, S.L., Hunker, J., Bulford, C.: Insiders behaving badly. IEEE Security and Privacy **6**, 66–70 (2008). DOI http://doi.ieeecomputersociety.org/10.1109/MSP.2008.87
15. Schwartz, N.D., Bennhold, K.: A trader's secrets, a bank's missteps. New York Times, 5 February 2009, New York, USA.
16. Probst, C.W., Hunker, J.: *The Risk of Risk Analysis-And its relation to the Economics of Insider Threats*, Proc. of the Eighth Workshop on the Economics of Information Security (WEIS 2009), June 2009.

Combatting Insider Threats

Peter G. Neumann

Abstract Risks from insider threats are strongly context dependent, and arise in many ways at different layers of system abstraction for different types of systems. We discuss various basic characteristics of insider threats, and consider approaches to the development and use of computer-related environments that require systems and networking to be trustworthy in spite of insider misuse. We also consider future research that could improve both detectability, prevention, and response. This chapter seeks to cope with insider misuse in a broad range of application domains—for example, critical infrastructures, privacy-preserving database systems, financial systems, and interoperable health-care infrastructures. To illustrate this, we apply the principles considered here to the task of detecting and preventing insider misuse in systems that might be used to facilitate trustworthy elections. This discussion includes an examination of the relevance of the Saltzer-Schroeder-Kaashoek security principles and the Clark-Wilson integrity properties for end-to-end election integrity. Trustworthy system developments must consider insider misuse as merely one set of threats that must be addressed consistently together with many other threats such as penetrations, denials of service, system faults and failures, and other threats to survivability. In addition, insider misuse cannot be realistically addressed unless significant improvements are made in the trustworthiness of component systems and their networking as well as their predictably trustworthy compositions into enterprise solutions — architecturally, developmentally, and operationally.

1 A Contextual View of Insiders and Insider Threats

We consider a broad spectrum of problems relating to insider threats, along with techniques for preventing, detecting, diagnosing, and understanding specific ex-

Peter G. Neumann
Principled Systems Group, Computer Science Lab, SRI International, Menlo Park, CA 94025-3493 USA, e-mail: neumann@csl.sri.com.

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_2,

ploits – within the context of overall system and application trustworthiness. For present purposes, an *insider* is simply a system user who is granted and can use certain privileges. Intuitively, insider threats involve such users misusing those privileges, potentially causing violations of confidentiality, data integrity, system integrity, system survivability, identity management, accountability, denials of service, and anything else relating to abuses of trust. For generality, we allow that the 'user' could also be some sort of human surrogate or other computer entity – process, agent, or system – presumably but not necessarily acting on behalf of specific human users.

The meaning of 'insider' is strongly dependent on the application context. Furthermore, the concept is relative to the privileges available (either given or somehow otherwise acquired), which may be hierarchically layered or otherwise granted. For example, in health-care applications, a doctor may have access to personal and medical data of many patients, whereas a patient may be able to access only certain portions of his or her own data—and nothing else. Administrators, insurance companies, pharmacies, and employers might have varied restricted access, as might nurses, third-party staff such as offshore transcription services, and surprisingly many others. In addition, banks, credit bureaus, data clearing houses, and many other people and institutions may have legitimate access to some medical information and related personal data. Furthermore, researchers and others might also have certain access rights, but possibly only with some sanitization or anonymization of personal data that might attempt to mask personal identities. Deborah Peel (patientprivacyrights.org) estimates that about 4 million people in the U.S. have some sort of access to medical record information—and thus they can all be considered as insiders in some respect.

Distinctions between insiders and outsiders can be slippery, particularly in the absence of system security that effectively reduces the likelihood of penetrations and external denials of service. Similarly, distinctions between malicious acts and accidental events are often misleading – in that events occurring accidentally could often be triggered intentionally, and adverse events may occur unbeknownst to the inadvertent triggerer. Clearly, a system often needs to be protected against both kinds of events. For example, most life- or safety-critical systems must address use and misuse by both insiders and outsiders in an architecturally integrated way, and enforceable policies must be established and consistently implemented. These subtleties can be quite significant in assessing how we should approach insider misuse within the more general context of system and network trustworthiness.

There are many different definitions of insiders, some of which are in conflict with one another. For example, some definitions exclude outsiders who have usurped privileges of insiders, whereas other definitions include those outsiders who have effectively gained privileges of insiders — perhaps due to inadequate system and process assurance. The latter individuals are herein referred to as *outside-inners*. Some definitions fail to define either 'insider' or 'outsider', simply implying that one is not the other. A definition attributable to a National Research Council study report is "a person who is allowed inside the security perimeter of a system and consequently has some privileges not granted outsiders." That definition suffers from defining one

type of user in black-and-white disjunction—as the opposite of the other—where neither is adequately defined and grey areas are ignored. It also suffers from the fact that today's computer operating system and networking infrastructures are vulnerable, and enable outsiders to carry out destructive integrity attacks almost as if they were insiders. It further suffers from the reality that there is typically no single security perimeter; indeed, at each layer of abstraction there may be different perimeters for different aspects of trustworthiness that satisfy different subsets of the set of given requirements, where compromises of one perimeter may directly or indirectly also compromise the integrity of other perimeters. Note that the different requirements for security, reliability, survivability, human safety, usability, and so on may be conflicting, and the respective perimeters of trustworthiness may actually be nonoverlapping – which suggests that the notion of an insider is actually multidimensional with respect to the different requirements and different types of applications. Furthermore, a supposed perimeter of trust may actually encompass the entire system, its total operational environment, all of its potential users, and perhaps the entire Internet – particularly in badly designed systems. The approach taken here generalizes other formulations into a multidimensional framework.

Misuse implies use that is contrary to expected operational behavior. However, that is too much of an oversimplification. In practice, the concept of *misuse* is meaningful only with respect to a policy that defines what usage is acceptable and what is not. Unfortunately, a basic gap exists between use that is intended to be acceptable and use that is actually possible (*e.g.*, [3]). Within that gap, subgaps exist – for example, between what is possible (because of design flaws and implementation bugs) and what is actually authorized, as well as limitations that result from inadequate granularity and expressiveness of access controls.

A useful distinction exists among three alternative misuse cases with respect to any particular layer of abstraction: *compromise from outside, compromise from within,* and *compromise from below.* Compromise from outside represents activities of outsiders relative to that layer of abstraction, but as noted above can result in outside-inners. Compromise from within represents actions of insiders with respect to that layer. Compromise from below represents actions of insiders with respect to some lower layer who can compromise the integrity of higher layers.

One historical example of how this trichotomy fits into the constructive architecture of an operating system is given by the Multics hierarchical ring structure [20], which provided eight layers of protection. That mechanism essentially ensured that each ring could not be compromised from higher layers, while the lowest-layer rings were under stringent development and operational configuration control. This mechanism also provided enhanced system survivability.

The determination of who is an insider and who is an outsider is also relative to what boundaries might be assumed to exist. That is, an insider at one layer may be an outsider with respect to a lower layer or with respect to a different perimeter. For example, someone who can manipulate bits in memory or secondary storage using hardware diagnostic tools might be called a hardware insider. Someone who can manipulate operating system parameters because she has authorized use of certain root privileges would be considered an insider with respect to the operating system.

Someone who can tamper with a browser because he is the maintainer of Web facilities would be considered an insider with respect to the webware. A similar analysis is given by Matt Bishop *et al.* [2]. For a recent compendium of articles on insider misuse, see [23]. See also the other chapters in this book, as well as Section 4 of the 2009 Roadmap for Cybersecurity Research [10], devoted to combatting insider threats. For some pithy examples of evident misuse, see the Appendix to this chapter (excerpted from [16]).

2 Risks of Insider Misuse

To understand the problems, we need to explore various kinds of insiders further, as well as types of misuse, threats, vulnerabilities, risks, and knowledge and experience that might be applied.

2.1 Types of Insiders

Differences among users may involve physical presence and logical presence. For example, there may be logical insiders who operationally are physically outside, and physical insiders who are logically outside. For present purposes, we consider both logical and physical insiders.

Clearly there are different degrees of logical insiders, relative to the nature of the systems and networks involved, the extent to which authentication and authorization are enforced, and the exact environment in which a user is operating at the moment. A user in one operational domain may be an insider at one moment and an outsider otherwise, with respect to each of the various so-called contexts noted above.

For example, if a system supports multilevel security (or multilevel integrity [1]), or even some form of multilevel availability or multilevel survivability [12]), then the existence of compartments suggests that a user can be an insider in one compartment but an outsider in another compartment, or an insider at Top Secret but an outsider with respect to all compartments. In that a user may operate at different levels and compartments at different times, the concept of insider is both temporal and spatial. In some sense, all users of a single-level Top-Secret system could be called insiders with respect to confidentiality, although they would appear to be outsiders relative to those others who were cleared into a particular Top Secret compartment. Similarly, a user could be an insider with respect to multilevel security and an outsider with repect to multilevel integrity. Thus, everything is relative to the frame of reference – what the user is trusted to be able to do, what privileges are required, what data or programs are being referenced, and whether the user authentication is strong enough to ensure that user identities are not spoofed.

With respect to conventional operating systems, database management systems, and applications functioning as single-level systems (even if lumping multilevel in-

formation into a single level, typically called *system high*), there are typically ordinary insiders who have passed the login authentication requirements and have been granted certain limited access rights. In addition, there are special users who are authorized to act as a superuser or otherwise be allocated extra-powerful privileges. In contrast, Trusted Xenix [7]) was a system in which the superuser privileges were extensively partitioned, where no one user holds all of the privileges, and where the granted privileges are insufficient to gain possession of all other privileges. (The iterative closure of static privileges augmented by privilege-changing privileges must also be considered whenever we consider what privileges are actually attainable by a given user or group of collaborating users.) In that rather ideal case, we might have no complete insiders, but many different types of relative insiders. Unfortunately, in the absence of meaningfully secure systems and fine-grained access controls that are properly defined, properly implemented, and properly administered, that ideal is still a fantasy.

Thus, we are confronted with a wide variety of insiders that is inherently multidimensional. Here, we tend to consider insiders somewhat loosely, avoiding fine nuances among different kinds of insiders. We assume that relative to a particular computational framework, insiders are users who have been authenticated to operate within that framework. However, where appropriate, we qualify that to include reference to the authorized privileges that may be specifically associated with a particular instance of an authenticated user (such as a system administrator).

2.2 Types of Insider Misuse

Along with the variety of insiders is associated a variety of types of insider misuse. One immediate categorization involves user intent, as in intentional versus accidental misuse (noted above). Even among intentional misuse, there is a wide range of possible actions – from outright malice to relatively benign annoyance, with many degrees in between. However, whether the cause is accidental or intentional is sometimes not clear.

A second categorization involves the evidential nature of the misuse, that is, whether the misuse is intended to be detected or hidden. System and network denials of service may be overt, in that they are readily obvious once they are enabled. However, stealthy Trojan horses that act as sniffers or that quietly leak information are typically intended to be covert, and may be intended to remain undetected as long as possible.

Although the focus here is primarily on intentionally malicious misuse, it is generally unwise to ignore accidental misuse. For example, the apparent success of what might be considered accidental but tolerated misuse could easily inspire subsequent malicious misuse. Furthermore, it is generally unwise to ignore stealthy forms of misuse. To the extent that detecting accidental misuse can be dealt with by the same mechanisms that are used for intentional misuse, accidental misuse need not be treated separately. Similarly, to the extent that stealthy misuse can be dealt with by

the same mechanisms that are used for more obvious misuse, stealthy misuse need not be treated separately – apart from possibly additional means of detecting it. Because seemingly accidental misuse may in fact be intentional misuse in disguise, stealthy misuse can be extremely dangerous; as a consequence, it is potentially risky to ignore any particular mode of insider misuse. Nevertheless, responses may differ depending on whether the cause is deemed to be accidental or malicious.

3 Threats, Vulnerabilities, and Risks

There are clearly differences in the nature of the various threats, especially in the possibility of outside-inners. Although an insider might conceivably have greater knowledge of the environment, and may thereby present greater threats, the differences between insider threats and outsider threats are often not stereotypically characterizable. If a system has meaningful authentication, many of the outsider threats can be made much less risk-prone, whereas most of the insider threats clearly remain. Also, firewalls that are well-designed, well-implemented, and well-configured can help somewhat, but today are also largely vulnerable to many attacks (such as active pass-through attacks using http, JavaScript, Active-X, PostScript, other forms of executable content, cross-site scripting, SQL injection, and bogus URLs in phishing attacks). The availability of meaningful additional authentication for insiders could be useful in inhibiting masquerading. With extensive monitoring, robust authentication may also help discourage misuse – especially if the identity of the perpetrator could be established and traced reliably. This may be especially relevant to insider misuse, if the true identity of the apparent user can be unequivocally determined (subject to exploitations of operating-system vulnerabilities – including manipulations of audit trails).

It is of course useful to consider insider threats in their own right. In today's systems, insider vulnerabilities and outsider vulnerabilities are both out of control. Serious efforts are needed to improve security and reliability of system and networks, and indeed to improve the overall survivability in the face of a wide range of adversities. With good external security in critical systems, insider risks may be much more serious than outsider risks. However, meaningfully precise access control policies and meaningfully secure fine-grained access controls may reduce the damage from the insider threats.

Table 1 itemizes some of the threats that appear to differ from outsiders to insiders. It ignores threats that are common to both outsider and insider perpetrators, such as carrying out personal attacks on individuals or corporations through an anonymous e-mail remailer, sending spams, creating monster viruses from a toolkit, creating risky mobile code, tampering with existing mobile code, intentionally crashing a system or component (although there are potentially mechanistic differences among insiders and outsiders), and so on. Nevertheless, to simplify the table, outside-inners are logically considered as outsiders unless they are knowledgeable enough to appear indistinguishable in something like a Turing-test sense from the insiders as

Table 1 Threats to Security

Attribute	Outsiders	Insiders
Authentication	Penetrations, attacks on PKI/authentication infrastructures, war dialing	Misuse of intended authority by over-authorized users, usurpation of superuser access and root keys
Authorization	Unprivileged exploitation of inadequate controls	Privileged manipulation of access controls
Confidentiality	Unencrypted password capture or compromise of encrypted passwords	National security leaks and other disclosures; access to crypto keys(!)
Integrity	Creating Trojan horses in untrusted components, Word macro viruses, untrustworthy Web code, in-the-middle attacks	Inserting Trojan horses or trapdoors in trusted (and untrusted) components; altering configurations, schedules, and priorities
Denials of Service	External net attacks, flooding, physical harm to exposed equipment	Disabling of protected components, exhaustion of protected resources
Accountability	Masquerading, DoS attacks on accounting infrastructures	Hacking beneath the audit trails, altering audit logs, compromising misuse detection
Other misuses	Planting pirated software on the Web	Running a covert business, insider trading, resource theft

whom they are masquerading – as might be the case with disgruntled recent ex-employees. More realistically, the indistinguishability may be more like the ability of an outsider to masquerade as an insider if just a little social engineering is all that is required.

3.1 Relevant Knowledge and Experience

Some differences are likely to exist in the knowledge available, the knowledge required, and the knowledge actually used in perpetrating various types of misuse. Understanding these differences may be useful in analyses associated with detected misuses.

For example, insiders might seem to have greater knowledge of what to look for in terms of sensitive information and particularly vulnerable programs in which to plant Trojan horses—including especially system administrators. In system-high systems, legitimate insiders are already likely to be gratuitously granted information to which they do not necessarily need access. In compartmented multilevel-secure systems, users would have clearances associated with authorizations, although that works both ways: a user not entitled to access a particular compartment is effectively an outsider with respect to that compartment, and indeed may not even know of the

Table 2 Knowledge Gained and Used

Outsiders	Ordinary Insiders	Privileged Insiders
Direct info and inferences from web info (such as penetration scripts), help files, social engineering; chats/ BBoards helpful	Experience gained from normal use and experiments; familiarity with sensitive files, project knowledge; collusion easy	Deep knowledge from experience; ability to change and abuse privileges; ability to create invisible accounts; collusion even easier

existence of the compartment if the system is properly implemented and operational procedures are properly enforced. However, users cleared into that compartment have an enormous advantage for potential misuse over users who are not – assuming isolation is suitably enforced and operationally deployed.

Table 2 makes a distinction among outsiders, ordinary insiders, and specially privileged insiders such as highly trusted system administrators, recognizing that we are lumping together users with common logical characteristics.

3.2 Exploitations of Vulnerabilities

There is a likelihood that an experienced insider can operate close to normal expected behavior (especially if engaged in a long-term effort at what in terms of a anomaly detection system would resemble statistical-profile retraining), which would be more difficult to detect. This increases the need for a variety of analysis techniques and correlation (see below).

Today, we have pervasive deficiencies in authentication, authorization, accountability, operating system security, network security, and intelligently deployed access controls. Given the absurdly poor existing state of the practical art of defensive security, the differences among exploitations by outsider and insiders may be less relevant than they would be in the presence of stronger security.

Insider exploitations might conceptually be thought of as somewhat simpler to manage in the presence of stronger system security. Enormous benefits could result from intrinsically better operating system security, network security, pervasive encryption, user authentication, and well-managed authorization. One of those benefits would be that detection and response could be much more precisely targetted, rather than having to address all security vulnerabilities. However, insider threats would still represent a significant problem, because many of those threats would not have been eliminated.

Table 3 Potential Severity of Risks Incurred

Outsiders	Ordinary Insiders	Extra-Privileged Insiders
Very serious in badly designed and poorly implemented systems, perhaps less serious with good user authentication and good auditing	Potentially very serious unless strong separation of roles, MLS, and fine-grained access controls; beware of system-high systems	Extremely serious, even with strong separation of roles and separation of privilges, MLS levels and compartments; misuse of multipurpose root privileges is inherently risky

3.3 Potential Risks Resulting from Exploitations

The potential risks may vary significantly from outsiders to outside-inners to ordinary insiders to highly privileged system administrators. However, it is in itself risky to give too much credence to these differences, because of several factors:

- When the security of systems, servers, firewalls, and networks is weak, both outside-inners and insiders can cause serious harm.
- Some outsiders such as terrorists may have highly visible major havoc in mind. Alternatively, outside-inners might try to mask the existence of clandestine Trojan horses, trapdoors, and other system aberrations. In general, exactly the same situation applies to insiders, although the stealthy route would generally be more likely in the presence of strong authentication that hinders insider masquerading and provides a fairly clear chain of evidence. Each type could create highly undetectable effects or massive disasters entailing major risks.
- Measures of risk are highly speculative, and strongly dependent on the application environment. (One man's feat is another man's poison.)

4 Countermeasures

Where possible, prevention is vastly preferable to detection and attempted remediation (although cases of insider misuse generally exist in which prevention is inherently difficult). For example, the Multics system architecture (see [5] and http://www.multicians.org/) stressed the importance of prevention by isolating privileged execution domains from less-privileged executions, isolating one user from another while still permitting controlled sharing (via access-control lists, access-checked dynamic linking, and dynamic revocation, as well as user-independent virtual memory), and using some sensible software-engineering concepts. Use of some of the Saltzer-Schroeder [22] security principles is directly relevant to minimizing insider misuse. The most obviously applicable principles here are separation of privileges, allocation of least privilege, and open design. In addition, ease of use (generalizing Saltzer and Schroeder's psychological acceptability) could provide incen-

tives for insiders to avoid the excuse of security being too complicated, which otherwise often results in the creation of unnecessary vulnerabilities. These and other principles are discussed further in the context of election systems in Section 7.

If there is no meaningful security policy, then the task of detecting and identifying deviations from that policy is not meaningful. If there is no fine-grained context-sensitive prevention in systems and networks, then even if there were a meaningful security policy, it would be difficult to implement it. With respect to insiders, enterprises operating within a system-high approach suggest that insider misuse is ill-defined – in the sense that everything may be permitted to all authenticated users. Thus, to have any hope of detecting insider misuse, we first need to know what constitutes misuse. Ideally, as noted above, it would then be much better to prevent it rather than to have to detect it after the fact.

The absence of rigorous authentication and constructive access controls tends to put the cart before the horse. For example, what does *unauthorized use* mean when almost everything is authorized? Recall the Internet Worm of 1988, which was an outside-inner attack. Robert Tappan Morris was prosecuted for *exceeding authority*; yet, no authorization was required to use the `sendmail debug` option, the `finger` daemon buffer overflow, the `.rhosts` mechanism, and copying an encrypted but then unprotected password file. This may have been *misuse,* but was not *unauthorized misuse*. The same issues arise with recent malware.

Finer-resolution access controls are of particular interest in minimizing insider misuse, such as fine-grained access-control lists and fine-grained roles, separation of duties, compartmentalized protection for integrity, and attribute-based encryption. Some of those controls date back to the Multics file system [6] in 1965, and have been the subject of refinement and alternative approaches ever since. Past work on strongly typed, hardware-tagged, capability-based systems (*e.g.*, [17]) could also considerably reduce opportunities for insider misuse; in such systems, access is impossible for anything for which an appropriate capability is not available. Of course, various forms of multilevel security and multilevel integrity could also help to narrow down the possibilities for insider misuse, albeit with the associated administrative baggage. However, all these approaches create further usability issues and administrative complexity.

Although relevant not specifically to insider misuse, but more generally to the development of trustworthy systems, several other thrusts are also of interest here – for example, a report on how to develop principled assuredly trustworthy composable architectures [13] and subsequent reflections on trustworthiness [14]. Also somewhat relevant is a paper by Paul Karger [8] that applies access controls to programs. That approach might be interesting in controlling the extent to which insider-introduced malware (particularly Trojan horses) could be blocked, assuming that the insider is not privileged to alter the access controls. Approaches to sandboxing have similar goals, limiting what would-be malware might be able to do.

In summary, better policies are needed establishing what threats are relevant, and what constitutes misuse. Better user authentication could not only prevent intruders from gaining insider access, but could also provide positive identification of insiders that might diminish their ability to masquerade as other insiders and to

otherwise hide their identities. Authorization is typically not fine-grained enough, which limits the effectiveness of access controls and misuse detection. Oversight and accountability are essential. Monitoring tools need to address detection of insider misuse.

4.1 Specification of Sound Policies for Data Gathering and Monitoring

Commericial products for misuse detection tend to assume a collection of known vulnerabilities whose outsider exploitations are associated with known policy violations. Existing products tend to be aimed primarily at penetrators and intrusion detection, and are not easily applied to detecting insider misuse. Policies for insider misuse tend to be strongly application-domain specific, and should dictate what is to be monitored, at what layers of abstraction. Thus, it is essential to have a well-defined policy that explicitly defines insider misuse, or else a policy that explicitly defines proper behavior and implicitly defines insider misuse by exclusion.

A much better understanding of the application domain is needed for monitoring users for potential insider misuse. Also, more detailed data may need to be collected. Furthermore, when someone is suspected of devious behavior, it may be desirable to go into a fine-grain analysis mode, although that has its own serious potential privacy problems.

Today, commercial systems for misuse detection generally rely on system audit trails, network packet collection, and occasionally physical sensors for their inputs. Other sources of input data are necessary for detecting insider misuse, including detailed database and application logs. In either case, the analysis systems need to obtain some knowledge of the perpetrator if they are to trace the detected misuses back to their initiators. In closed environments, there can be much better user authentication than in open environments, although masquerading is still possible in many operating systems and application environments. Whenever that is the case, the actual choices of data to be gathered for insider-misuse detection tend to differ from that of intrusion detection. However, the existence of logical insiders who are physically outside and logical outsiders who are physically inside may make such distinctions undesirable – suggesting that making the assumptions (such as there are no outsiders, or there is no insider misuse) is unwise. The necessary use of encryption for stored information in highly sensitive systems may also complicate the gathering of information on potential insider misuse, and necessitate capture of unencrypted content – which raises serious some serious security and privacy concerns.

4.2 Detection, Analysis, and Identification of Misuse

In the absence of good prevention, it is of course desirable to detect known defined types of misuse (*e.g.*, through rule-based detection) as well as otherwise unknown types of anomalous misuse (*e.g.*, seemingly significant deviations from expected normal behavior). The latter type of detection could be particularly important in identifying early-warning signs of misuse. o Because there are potential differences in the data that may need to be collected, there may be some differences in the approach to detection of misuse among the different types of misuse, depending on the relative roles of insiders and insider misuse. If insiders can exist only within local confines (for example, as in the case of a multilevel security compartment in a system with no remote users and no Internet connectivity), it may be unnecessary to collect packets and other network data – which themselves constitute potential security and privacy risks. If privileged logical insiders are also able to access their systems remotely (for example, using encrypted programs such as `ssh` from outside) and are in some sense then indistinguishable from outsiders at least geographically or from their external Internet presence, then networking data may also be relevant. Clearly, the presence of strong authentication has an impact on carrying out insider misuse detection.

Similarly, there may be differences in data retention requirements among misuse-detection system. If the intent is to gather sufficient information to prosecute insider misusers, then the situation is quite different from detection whose aim is merely to detect the presence of misusers so that other extrinsic methods (such as wiretaps, cameras, and physical surveillance) can be invoked. (These differences may also apply to outsiders – although the relative priorities are likely to be different.) In general, long-term retention of raw audit logs and of digested (analyzed) data is recommended.

The marketplace for intrusion detection is aimed primarily at detecting known attacks by outsiders – for example, with signature-based expert systems seeking to detect exploitations of known vulnerabilities. The idea of rapidly deploying an analysis system is meaningful for a given firewall, or for a given operating system, or for a given application for which a set of rules have already been written. However, insider attacks tend to be much more domain specific; insider analysis requires more detailed analysis of the threats and risks, some skilled implementation of rules, judicious setting of statistical parameters, and some further work on analysis of the results. These rules must also take into account discrepancies between the actual access controls and what kind of access is considered appropriate. Once again, the extent to which explicit fine-grained access controls can be defined and enforced has a direct influence on what kinds of insider misuse need to be detected. Thus, new approaches are needed to better address the insider threats. This is not a straightforward off-the-shelf installation process.

In a multilevel compartmented system/network environment, in which there are presumably no outsiders and in which the insider threat predominates, monitoring and analysis take on multilevel security implications, with many opportunities for covert channels and inferences. Monitoring can be done compartmentally, but aggre-

gation, higher-level and cross-compartment correlation on an enterprise-wide basis present serious potential multilevel security problems.

More emphasis is needed on not-well-known forms of insider misuse, on interpretation of detected anomalies, and hierarchical and distributed correlation. Much more emphasis is needed on tools to aid in the deployment and configuration of analysis tools for domain-specific applications. Serious effort might also be devoted to multilevel-secure analysis (and response) in contexts in which MLS systems might be important. Procedural and psychological approaches are likely to predominate, much greater awareness of the threats and risks of insider misuse is likely to drive new approaches.

An enormous risk exists relating to false accusations of supposed culprits despite the inability to carry out any definitive traceback to host systems and individual logins. This problem is exacerbated by the long-time retention and undeletable mirroring of erroneous data throughout the Internet, and the difficulties in correcting widely disseminated erroneous information (as was the case in the 'Swift Boating' of John Kerry in the 2004 U.S. Presidential election campaign).

4.3 Desired Responses to Detected Anomalies and Misuses

In some cases of outsider attacks (particularly denials of service), it is more important to stave off the attacks than to let them continue. In other cases, it may be appropriate to let the attacks continue but to somehow confine their effects (as in the case of sandboxing and honeypots). A similar range of responses exists for insiders. In some cases of insider misuse (particularly where the perpetrator has been identified and prosecution is anticipated), it may be particularly important to detect the misuse, to allow it to continue (perhaps under special system constraints and extended data gathering such as key-stroke capture), and monitor it carefully – without giving away the fact that detailed surveillance is being done.

Thus, there are clearly differences in the desired responses that may be considered once misuses have been detected. However, the full range of possible responses may also be applicable to both insiders and outsiders – although possibly in different degrees in the two cases. In any case in which continued misuse is allowed, serious risks exist that undetected contamination and other integrity problems may occur and remain subsequently. This must be factored into any dynamic strategies for real-time response to detected misuse.

5 Decomposition of Insider Misuse Problems

This section looks at insider misuse in the context of the bigger picture of security. It considers development and operation, and the effects of those issues on misuse

detection. It also specifically addresses the importance of user profiling and the desirability of extending it to include psychological factors.

5.1 Stages of Development and Use

Each of the stages in system development and use has its own problems and potential vulnerabilities that must be considered with respect to insider threats. Various system development methodologies address some of these problems. (For example, see [13, 14].)

- Requirements: Insider threats are often ignored, even in highly sensitive standalone systems that tend to run at system high. Security, reliability, survivability requirements are often short-sighted, incomplete, or unsatisfied in system developments.
- System architecture and design: Many commercial systems are short-sighted, hindered by their needs for backward compatibility with earlier nonsecure systems and networking, and handicapped by a serious lack of commitment to robustness. These systems are primarily aimed at low-hanging fruit, or else must have their rule bases updated frequently to keep up with the malware du jour. In contrast, the research community has progressed significantly in recent years. For example, [18] discussed some of the research directions as well as the desirable characteristics of future systems for anomaly and misuse detection that could be applicable to insider misuse as well as intruders.
  System design must encompass authentication and authorization that is relevant to the insider threats. Authentication can seriously impede outsiders, but not if the systems rely on fixed passwords (especially if those passwords are transmitted unencrypted, or are replayable, or used in single-signon applications across boundaries of trustworthiness). Authorization is typically not fine-grained enough, and limits the effectiveness of access controls and misuse detection. Boundary controllers may be useful, but are typically vulnerable to denial-of-service attacks.
- Implementation: Many serious security-related implementation flaws persist. As just one example, buffer overflows continue to appear despite years of knowledge of their origins and the ensuing risks. Serious attention to software engineering discipline is sorely lacking.
- Operation: Even with ideal system development, accountability is fundamental to monitoring and analyzing insider misuse. It needs to be tightly coupled with strong authentication and access controls.
- System administration: System administrators are hard-pressed to cope with security flaws, security patches, and administering misuse detection. System design must include more effective mechanisms to aid admins, such as self-configuring detection and analysis tools that can be easily tuned to the threats of greatest significance according to the perceived risks.

- High-level enterprise management: Existing analysis techniques do relatively little for enterprise-wide monitoring and correlation across multiple network and system platforms.
- System support: Although vendors may not always have the customer's best interests at heart, many customers seem to be overly naive. In an insider misuse workshop on 12 April 1999, Ed Amoroso mentioned AT&T's experience with Net Ranger (later acquired by Cisco). When he and his colleagues finally tried to install it, months after receiving the CD, they discovering that certain files were missing from the installation CD. When they complained to Cisco, the Cisco folks indicated they had never before heard about this problem; apparently no one had ever successfully installed it!
- Monitoring: Data from appropriate audit trails (operating systems, DBMSs, applications) and network data (packets, network management information) needs to be hierarchically abstracted, heterogeneous, diversified, and collected only where needed for potential privacy reasons. Existing data sources tend to have little abstraction and contain huge quantities of relatively useless information.

Detection of insider misuse (and especially hitherto unrecognized threats) deserves much greater attention, using a wide variety of approaches. Historically, two early SRI efforts specifically aimed at detecting insider misuse are worth noting. The first, begun in 1983 for the CIA, sought statistically significant deviations from the expected normal behavior of IBM mainframe users represented in user profiles. Later, in the 1990s, a variant of the NIDES system was considered for the classified FBI Field Office Information Management System (FOIMS), using a rule-based expert system applied to database logs. Neither system was actually deployed, perhaps because each institution decided that insider threats were realistically minimal! (Note the first bulleted item in the Appendix.)

Misuses and anomalies that have been detected require some abstraction in their reporting and diversity of sites to which reporting occurs. (For example, EMERALD [19, 18] allowed for a wide variety of destinations, including passing the results to higher-layer instances of EMERALD and directly to system administrators.) Correlation is needed across wider scopes across multiple target systems and networks, and across multiple analysis platforms. Much more sophisticated and understandable interpretation of analysis results is essential at varying layers of abstraction.

5.2 Extended Profiling Including Psychological and Other Factors

It is clear from the above discussion that detecting insider misuse must rely heavily on user profiling of expected normal behavior (although some use can be made of application-specific rules). Efforts to date have concentrated on relatively straightforward statistical measures, thresholds, weightings, and statistical aging of the profiles, independent of particular users. Considering that much is already known about insiders, it would seem highly desirable to include additional information in

the profiles. Physical attributes might include access to buildings, rooms, and computers based on real-time access records, using badges, logins to local machines, biometric authentications, interactive pager probes, cameras, and other sensor data. Planning data might include expected activities such as travel schedules, special arrangements regarding working hours, and facility restrictions. The combination of physical whereabouts and expected whereabouts could also be used to detect stolen badges or stolen authentication information of people who are highly trusted.

In previous systems aimed at insider misuse as well as intruders, statistical profiling (*e.g.*, in NIDES and EMERALD) provided the capability of monitoring individualized computer activities, such as which editors the user prefers, which programming languages, which mail environment, which variants of commands, and so on. This approach seems to be less relevant today, but still has potential where intrusion is not a primary concern.

Personal on-line behavior can also be profiled statistically by extending the analysis information that is recorded, such as with whom an individual tends to exchange e-mail, which Web sites are visited regularly, and even what level of sophistication the user appears to exhibit. There are also biological factors that might be monitored, such as how often a user gets up for breaks (activities that could also be monitored by physical access controls).

In environments in which monitoring key strokes is not considered intrusive, some effort has been made to monitor key-stroke dynamics. This approach tends to be much less reliable in general, particularly with confronted with network and satellite delays. Also, if you are typing with one hand because you are drinking a cup of hot coffee with the other hand, your typing dynamics are of course specious.

In addition to providing a real-time database relating to physical whereabouts, and extending statistical profiling to accommodate subtle computer usage variants, it would also be appropriate to represent certain external information regarding personal behavior, such as intellectual and psychological attributes.

As an example of an intellectual attribute, consider writing styles. There are already a few tools for analyzing natural-language writing styles. Profiles of individual-specific "msipelings", the frequency of obscenities and the choice of explicit expletives, the relative use of obscure words, and measures of obfuscational proclivities and Joycean meanderings might also be quite useful. (Recall Tom Lehrer's warning: Don't write naughty words on walls if you can't spell.)

Psychological factors do not seem to have been explored much in the past, especially in the context of insider misuse. Psychologists routinely observe certain standard behavioral characteristics and analyze deviations therefrom. Some of those characteristics that are particularly relevant to potential insider misuse might be modeled in extended user profiles. As one specific example, we might be able to develop measures of relative paranoia, based on how often a particular user invoked certain commands to observe who else might be observing what that user was doing in real time, or the use of aliases in posting to newsgroups. A measure of aggressive behavior could be interesting, but would probably require some human reporting of perceived relative hostility levels in e-mail messages received from a given individual. Measures of anger and stress levels in general computer usage could also

be conceived. However, considerably more effort is needed to characterize which psychological attributes might be effectively utilized. However, this is not likely to have much success, because there are not well established characteristics, and human variabilities are likely to confound them anyway.

If this approach is considered possibly fruitful, we should approach some psychologists who are familiar with computer users and ask them to speculate on psychological factors that might be both computer detectable and behaviorally discriminative with respect to insider misuse. On the other hand, users tend to be not particularly inherently risk-aware. However, see a *CACM* Inside Risks column by Dr. Leonard Zegans [25], which observes that, with respect to computer technology, users tend to take risks unconsciously and in many cases unwillingly.

The concepts noted above have significant privacy implications that must be addressed.

6 Requirements for Insider-Threat-Resistant High-Integrity Elections

The general problem of dramatically increasing the integrity of computerized election processes is in some sense a paradigmatic hard problem that encompasses a wide diversity of requirements addressing voter privacy, system integrity, data integrity, data confidentiality, system survivability, accessibility, and other issues. This problem nicely illustrates that the notion of an 'insider' is highly context dependent. sequentially and hierarchically varying, and distributed.

Insider threats exist essentially in every phase of election processes, including before, during, and after voting actually occurs. Within the development and use of voting technology, insider threats abound among system designers and implementers, system purveyors, system administrators, workers storing or installing voting machines, poll workers, poll judges, election officials, and anyone else with physical or logical access to voting systems – including voters who may be able to introduce Trojan horses or compromises (*e.g.*, through altered or privileged access cards). Note that anyone with physical access may be an insider in certain respects, but ideally would seem to be an outsider with respect to altering software, ballot definitions, data, and so on. Unfortunately, existing systems are sufficiently flawed that almost anyone could be a potential insider. Insider threats also exist among election officials and government employees extrinsic to and essentially independent of any technology,

Requirements for trustworthiness appear in various guises across the entire spectrum of system development and operation. They are relevant to electronic systems, but also to some extent to paper-based and mechanical lever systems. They apply irrespective of whether computer technology is used to prepare ballots, to tabulate ballots, or just to compile results – or indeed not at all (which is increasingly rare in the U.S., but quite common in other countries). They must also include oversight of people, on whom the systems and procedures are ultimately heavily dependent.

Risks of insider misuse arise whenever such a requirement is not satisfied (*e.g.*, [16], click on Election Problems). Suggested but by no means complete integrity requirements are summarized in the following stages. Ideally, end-to-end assurances of integrity, security and privacy must span all of these stages.

- Voter registration. Prospective voters must be impartially vetted for eligibility. Voter database entries for all eligible registrants must be timely and correct, tamper evident, and carefully audited for misuse; database errors must be detected as soon as possible, and be much easier for voters to correct quickly when wrong or not up to date. Accidental and intentional disenfranchisement of legitimate voters must be avoided, with voters assumed innocent until proven guilty.
- Voter authentication and authorization. Voter identification and validation must be impartial. Dependence on erroneous databases to validate users must be avoided, and errors easily corrected. Requisite alternative databases or paper backups must be available during elections in case primary sources are unavailable. Disputed challenges must result in voters casting provisional ballots, which must be fairly and promptly counted if valid. (Experience with the the Employment Eligibility Verification System (EEVS) [15] and its successor US-VISIT suggest that database errors present some enormous problems.)
- Voter information. Voters must be officially notified in a timely manner that is clearly differentiated from bogus information (exemplified by last-day phone calls informing selected voters that their polling places had been changed, or that they should vote on Wednesday to reduce polling place congestion, as occurred in November 2008).
- Polling place availability and accessibility. Adequate voting machines, paper ballots, provisional ballots, and so on must be available for all polling places. Past experiences with long delays in certain precincts need to be avoided. Alternative methods must be available (*e.g.*, paper ballots) when machines are inoperable. Early voting and absentee voting both should be suitably vetted and tamper evident; they present many problems of their own, and require careful oversight. (They must also be based on voter convenience, rather than constrained by draconian rules.) Disabled and disadvantaged voters must be allowed to vote conveniently and without undue time pressures from poll workers.
- Ballot layout and allocation. Ballot formats must be easily understood, unambiguous, and nonbiased, preferably thoroughly vetted beforehand. Voters must be given correct ballots (*e.g.*, in the proper language, and for the correct precinct, especially in multiprecinct polling places).
- Vote casting. Erroneous and poorly designed ballot faces and machines with confusing human/system interfaces must be avoided. Machines with identified fundamental flaws (such as the lack of audit trails and observed vote flipping) must not be used. The voting process must be voter friendly. Votes must be correctly recorded, and that process must be demonstrably verifiable.
- Vote counting and canvassing. The tabulation of votes must be *demonstrably* correct, with suitably high probability, reproducible through independent cross-checking, and demonstrably not subject to accidental errors, manipulation, and

Table 4 Insider Threats to Elections

Functions	**Intrinsic Insider Threats to Voting Systems**
Authentication	Misuse of various administrator and other login privileges; superuser usurpation; undocumented backdoor passwords
Access controls	Exploitation of granted privileges and flaws in voting software, underlying operating systems, and hardware
Confidentiality	Access to individual votes and crypto keys and passwords, before, during and after elections
Integrity	Tampering with voting machines and back-end tabulating systems: inserting bogus votes; altering existing votes; inserting trapdoors, Trojan horses, and -in-the-middle attacks
Service denials	Disabling protected components, resource exhaustion; power disruptions; rendering systems unbootable or otherwise unusable
Accountability	Altering audit trails and misuse detection subsystems (if any!)
Roles	**Extrinsic Insider Threats to Election Procedures**
Registration	Inserting bogus registrations, disenfranchising legitimate voters
Vetting voters by poll workers	Authorizing bogus voters, disenfranchising legitimate voters (*e.g.*, requesting *three* pieces of identification, or none, or providing a ballot for the wrong precinct)
Assistance	Misleading voters, *e.g.*, with erroneous instructions
Canvassing	Inserting bogus ballots, altering existing ballots or totals
Monitoring	Elections in which one party appoints both poll checkers
Remediation	Tampering with recounts, forcing do-overs

unmonitored tampering. The authentication of the totals must be essentially incorruptible.

- Monitoring. Oversight and visibility are essential. Computer audit trails and paper logs must be sufficiently comprehensive to allow for detection of errors, manipulation, and tampering, and valid for forensic use if needed. Insider denials of integrity and fraud need to be easily detected.
- Remediation. Following detected irregularities, meaningful definitive recounts must be possible via independent means (with mandated do-overs whenever sufficient evidence of uncertainty warrants). Random partial recounts must be mandatory, with greater extent for closer elections. Remediation itself must be free of errors, manipulation, and tampering.
- Openness throughout. All the above stages in the election process must be subject to thorough scrutiny and oversight. End-to-end assurances that these requirements are satisfied should be sought, encompassing registration, voter authentication, voting, overall accountability, and the resulting stages. Any gaps in the end-to-end assurance are likely to present opportunities for compromise.

An obvious conclusion from this brief itemization is that the insider threats are ubiquitous and pervasive. Every step in the election process is a potential weak link that can be easily exploited or accidentally exercised by insiders – especially in the absence of thorough monitoring and audit trails. When computers and electronic

communications are involved, the risks may be greater than in purely manual election processes. However, insider threats are significant in any case.

Table 4 summarizes some of the main threats from insiders. In this context, outside-inners are considered as insiders.

7 Relevance of the Countermeasures to Elections

Reflecting on the paradigmatic nature of the election problems, it is useful to consider how the countermeasures suggested above might apply to insider threats to elections. In that context, two sets of principles are particularly relevant: (1) Saltzer and Schroeder [22] as reformulated by Saltzer and Kaashoek (see Chapter 11 of [21] as well as some earlier extensions and reinterpretations [13]), and (2) Clark and Wilson [4]. These principles are paraphrased below for present purposes. (Clearly, some of these principles are relevant to outsiders as well as insiders.)

Saltzer-Schroeder-Kaashoek (SSK) Security Principles

- Economy of mechanism. Having to trust very complex systems is typically risky, especially in the presence of insiders. Complex mechanisms are inherently more likely to have vulnerabilities. Complex voting software that depends on a large underlying operating system with known vulnerabilities and many potential insiders engenders compromise from below through the operating systems and compromise from within through alterations of the voting software and its data.
- Fail-safe defaults. Access controls should generally permit access explicitly with a default of no access, rather than a default of total access unless explicitly denied. This simple-sounding principle can considerably reduce the likelihood of insider misuse, especially when combined with the other principles. Overly permissive access controls are inherently risk-prone. And yet, many of today's voting machines have few internal controls.
- Complete mediation. Whatever security controls are in place, they should not be bypassable or subvertible. This principle applies especially to monitoring and the completeness and integrity of audit trails.
- Open design. Reliance on the secrecy of a design and proprietary source code is generally a bad idea. Yet, security by obscurity pervades the marketplace for electronic election systems. The commercial voting system vendors in the U.S. persist with proprietary ownership of their software, its evaluations (which are commissioned and paid for by the vendors), and the internal data formats and election data. The notion that security would be diminished if anyone else had access to that proprietary knowledge is clearly a myth. However, the 'many eyeballs' approach must be countered by strict adherence to system integrity – which requires stringent version configuration control.
- Minimization of secrets. Closely related to the principles of open design and economy of mechanism is the notion of minimizing what must be trusted – and

indeed what must be trustworthy. For example, the confidentiality of cryptography necessarily depends on the protection of the crypto keys, but should not have to depend on the secrecy of the algorithm. In general, system architectures that strongly minimize what must be trustworthy are preferable to systems that do not. As an example, Ka-Ping Yee's Ph.D. thesis [24] on the Pvote electronic voting systems shows how the amount of software that must be trusted for election integrity can be dramatically reduced – to 460 lines of Python (not counting the Python execution environment, underlying operating system software, and hardware).

- Separation of privileges. Privileges should be defined separately for different roles, such as system developers, system operators, contractors, election officials, and auditors who attempt to resolve discrepancies, as well as authorized voters and provisional voters. Those privileges must be explicitly associated with needs to prevent unauthorized access to software, data, and system configurations. (See the principle of least privilege.)
- Least privilege. Given separation of privileges, only necessary privileges should be allocated according to appropriate roles. For example, voters should have no system privileges other than the ability to cast their votes. Vendors should not be permitted to alter certified code prior to or during an election (even if there is a complete audit record of what they have done and why). Vendor contractors and election officials who set up ballot faces should not be permitted to make any alterations to system configurations, software, or election data. Before and after elections, officials may have to perform some carefully controlled and audited reconfigurations – such as initializing an election, or closing it down. Vendors, contractors, and election officials should not be able to cast or alter any votes when serving in those roles. Test programs and test results should have absolutely no effect on the live election results. Voters should be unable to modify votes of other voters – or system configurations. And so on.
- Least common mechanism. Building a voting system on top of a widely used operating system that is also used for other purposes (such as system development and debugging) is not particularly wise. (However, using a lesser-known operating system still relies on security by obscurity—see above.) That approach can greatly increase the opportunities for insider misuse by anyone with access to the operating system. Similarly, system development and debugging should not be permitted on live systems. Furthermore, simplistic suggestions that ATMs (which identify and authenticate customers individually, with detailed audit trails and time-stamped photographic images) could also be used for voting (where anonymity and total privacy are desired) would most likely result in compromises of both banking integrity, vote integrity, and voter privacy!
- Ease of use. Where systems are not easily used, voters, poll workers, and others will make mistakes. Where systems are not easily maintained and are difficult for election officials to configure correctly with respect to security controls (or the lack thereof), certain responsibilities are likely to be outsourced to system vendors, such as ballot-face preparation, pruning supposed convicted felons from voter lists based merely on partial name matches, and oversight of the integrity

and ultimately the authority of elections. Clearly, these and other practices can increase opportunities for insider and even outside-inner misuse. Considerable difficulties are also created by systems that are supposedly useful for disadvantaged voters.

- Pervasive auditing. Stringent auditing within trustworthy systems could enable meaningful recounts in cases of disputes (a facility that is almost nonexistent at present) and further provide forensic-quality evidence in cases of tampering or accidental changes. Such auditing should achieve high integrity, completeness, nonalterability, and nonsubvertibility, and ideally must be derived independently from the systems being monitored. Redundancy is particularly important here. (This is a generalization and strengthening of what is termed 'compromise recording' by Saltzer and Schroeder.)

Clark-Wilson Integrity Model

For completeness, the Clark-Wilson properties are enumerated here in a simplified form, even though there is some overlap with the Saltzer-Schroeder-Kaashoek principles, and some of them are less relevant specifically to insider misuse.

- Enforcement 1: Users may operate on data only through controlled operations, never directly (SSK: Encapsulation, complete mediation). Back-door access to programs and data, reset commands, and alterable audit trails are risky – before, during, and after elections!
- Enforcement 2. Users may perform operations only if explicitly authorized (Authorization).
- Enforcement 3. User identities must be authenticated (Authentication).
- Enforcement 4. Authorizations can be changed only by a security officer (Nondiscretionary controls). However, those privileged users clearly remain an insider threat.
- Certification 1. Data must be verified as a consistent representation of the real world (Data validation). This would also tend to flag situations in which the number of votes exceeds the number of voters.
- Certification 2. Programs must implement operations as *well-formed* transactions (Consistent transformations). Transactions need to be certified before going into operation. Insider manipulations of voting data could be detected via inconsistencies in cross-checking or violations of integrity seals.
- Certification 3. Systems must enforce separation of duties. (SSK: Roles, separation of privilege, and least privilege are fundamental to coping with insider threats in voting systems.)
- Certification 4. All operations must be audited. (SSK: Complete auditing)
- Certification 5. Operations must validate inputs, or else reject them (Input validation and atomicity). An untrusted operation must still be atomic – that is, either completely accepted or completely aborted.

Most of the Saltzer-Schroeder-Kaashoek and Clark-Wilson principles can contribute significantly to election integrity in the face of accidental and intentional insider misuse, along with other principles (*e.g.*, [13]). For example, the principle

of separation of policy and mechanism suggests that policy that may need to be altered should not be embedded in mechanism. A rather astounding example of the violation of that principle is found in Sequoia's election systems that are reprogrammed *after the system is certified* in order to support each different ballot face. For Diebold (now Premier), in every one of 17 counties using their equipment in the California election of November 2003, the versions of those systems that were in operation were not the certified versions. Such violations of what is a common principle in software engineering represent one of the most dangerous examples of opportunities for insider misuse in existing voting machines.

Overall, election officials today have only superficial control over the entire life cycle, including operations. The complexity of some of the all-electronic systems is such that the major vendors tend to provide their own personnel to help with setting up ballot faces and addressing technical problems that occur before, during, and even after elections. In the existing commercial systems, developers and vendors have considerable latitude in making surreptitious system changes that could alter the results of elections—including cases in which election software was not the certified software (as noted above). The absence of meaningful audit trails and the alterability of audit trails that do exist further complicate the process, because of the lack of demonstrable provenance and an almost complete lack of records on the alteration of code and election data. Ironically, the Nevada Gaming Commission and other states hold gambling systems to standards and evaluation processes that are astoundingly more stringent than those for voting systems.

As is the case elsewhere, various approaches to coping with insider threats may also be applicable to coping with outsider threats, and in some cases to natural disasters and other so-called acts of God. Once again, we stress the importance of system architectures and development processes that are capable of systematically addressing all of the critical requirements – including usability.

8 Research and Development Needs

Overall, there is still much research and development work to be done, some useful for trustworthiness in general, some specific to insider threats, and some specific to elections.

- The commonalities among insider and outsider misuses need to be understood, with respect to threats, methods, exploitations, detection techniques, and responses, taking advantage of those commonalities where possible, and resorting to different but compatible approaches where commonality is not immediately evident – all within the context of developing trustworthy systems that can address insider misuse as well as other critical requirements. Significant effort must be devoted to defining characteristic types of insider misuse. (For example, see [9, 11].)
- Finer-grained access policies and access controls are needed to help define what constitutes proper usage, thus facilitating the role of insider-misuse detection.

Greater effort needs to be devoted to detecting unknown modes of misuse, rather than focusing on just detecting known attacks. Hierarchical and distributed correlation is also required, to identify common patterns and user intentions. Extrinsic individual characteristics such as psychological behavior that might be included in profiling user activities need to be identified.

- Better system architectures to minimize what must be trustworthy (*e.g.*, Pvote [24]) and better software engineering practices are needed for developing high-integrity trustworthy computer systems (including subsystems for monitoring and analysis) to make them more composable [13] and interoperable (to permit mix-and-match multivendor systems), robust, evolvable, and extensible in their application domains—including attributes such as reliability, fault-tolerance thresholds, survivability, performance, and suitability for combatting insider misuse. For example, the NSF ACCURATE center (http://accurate-voting.org) is exploring various alternatives, including the development of VoteBox and experimentation with its usability (http://votebox.cs.rice.edu), Pvote, and other efforts.
- Real-time analytic systems must themselves be tamper resistant, to hinder integrity and denial-of-service attacks, alterations of evidence (either by malfeasors to cover their tracks, or by law enforcement in attempting to fake evidence).

9 Conclusions

In general, insider misuse cannot sensibly be treated as an isolated problem. For example, combatting it critically depends on the existence of meaningfully trustworthy systems and applications, nonspoofable identity management and user authentication, fine-grained access controls, controlled monitoring that is not excessively privacy invasive, carefully specified and enforced operational practices, transparency in the sense of being readily understood, privacy-respecting oversight, metrics for determining effectiveness of countermeasures, and ultimately the honesty, integrity, and diligence of trusted users. Although most of these approaches are relevant more pervasively, they are particularly important in elections – where total-system approaches and end-to-end assurances are essential in monitoring and protecting against insider threats.

In the spirits of Saltzer-Schroeder-Kaashoek and Clark-Wilson, this chapter may seem to be old wine in new bottles. However, the vintage is superb and still timely. Besides, still wine runs deep – the spirits are willing, but the flash (memory) is weak.

Acknowledgment. This chapter is based on a position paper, Combatting Insider Misuse, with Relevance to Integrity and Accountability in Elections and Other Applications, for the Dagstuhl Workshop on Insider Threats, 20-25 July 2008, It was prepared with funding from National Science Foundation Grant Number 0524111, under the SRI project for ACCURATE: A Center for Correct, Usable, Reliable, Auditable and Transparent Elections.

The author is grateful to Matt Bishop, Drew Dean, Jeremy Epstein, and Sean Peisert for some incisive interactions. Matt also presented my position paper at the Dagstuhl 2008 workshop.

References

1. K.J. Biba. Integrity considerations for secure computer systems. Technical Report MTR 3153, The Mitre Corporation, Bedford, Massachusetts, June 1975. Also available from USAF Electronic Systems Division, Bedford, Massachusetts, as ESD-TR-76-372, April 1977.
2. M. Bishop. Position: 'Insider' is relative. In *Proceedings of the 2005 New Security Paradigms Workshop*, pages 77–78, Lake Arrowhead, California, October 2005.
3. M. Bishop, S. Engle, C. Gates, S. Peisert, and S. Whalen. We have met the enemy and he is us. In *Proceedings of the 2008 New Security Paradigms Workshop*, Olympic Valley, California, 2008.
4. D.D. Clark and D.R. Wilson. A comparison of commercial and military computer security policies. In *Proceedings of the 1987 Symposium on Security and Privacy*, pages 184–194, Oakland, California, April 1987. IEEE Computer Society.
5. F.J. Corbató. On building systems that will fail (1990 Turing Award Lecture, with a following interview by Karen Frenkel). *Communications of the ACM*, 34(9):72–90, September 1991.
6. R.C. Daley and P.G. Neumann. A general-purpose file system for secondary storage. In *AFIPS Conference Proceedings, Fall Joint Computer Conference*, pages 213–229. Spartan Books, November 1965.
7. V.D. Gligor *et al*. Design and implementation of Secure Xenix[TM]. In *Proceedings of the 1986 Symposium on Security and Privacy*, Oakland, California, April 1986. IEEE Computer Society. also in *IEEE Transactions on Software Engineering,* vol. SE-13, 2, February 1987, 208–221.
8. P.A. Karger. Limiting the damage potential of discretionary Trojan horses. In *Proceedings of the 1987 Symposium on Security and Privacy*, pages 32–37, Oakland, California, April 1987. IEEE Computer Society.
9. C.E. Landwehr, A.R. Bull, J.P. McDermott, and W.S. Choi. A taxonomy of computer program security flaws, with examples. Technical report, Center for Secure Information Technology, Information Technology Division, Naval Research Laboratory, Washington, D.C., November 1993.
10. D. Maughan *et al.* A roadmap for cybersecurity research. Technical report, Department of Homeland Security, November 2009.
11. P.G. Neumann. *Computer-Related Risks*. ACM Press, New York, and Addison-Wesley, Reading, Massachusetts, 1995.
12. P.G. Neumann. Practical architectures for survivable systems and networks. Technical report, Final Report, Phase Two, Project 1688, SRI International, Menlo Park, California, June 2000. http://www.csl.sri.com/neumann/survivability.html.
13. P.G. Neumann. Principled assuredly trustworthy composable architectures. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, December 2004. http://www.csl.sri.com/neumann/chats4.html, .pdf, and .ps.
14. P.G. Neumann. Reflections on system trustworthiness. In Marvin Zelkowitz, editor, *Advances in Computers, volume 70*, pages 269–310. Elsevier Inc., 2007.
15. P.G. Neumann. Security and privacy in the employment eligibility verification system (eevs) and related systems. In *Congressional Record*, Washington, DC, Jun 7 2007. U.S. House of Representatives.
16. P.G. Neumann. Illustrative risks to the public in the use of computer systems and related technology, index to RISKS cases. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, 2009. Updated now and then:

http://www.csl.sri.com/neumann/illustrative.html; also in .ps and .pdf form for printing in a denser format.
17. P.G. Neumann, R.S. Boyer, R.J. Feiertag, K.N. Levitt, and L. Robinson. A Provably Secure Operating System: The system, its applications, and proofs. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, May 1980. 2nd edition, Report CSL-116.
18. P.G. Neumann and P.A. Porras. Experience with EMERALD to date. In *Proceedings of the First USENIX Workshop on Intrusion Detection and Network Monitoring*, pages 73–80, Santa Clara, California, April 1999. USENIX. Best paper.
19. P.A. Porras and P.G. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *Proceedings of the Nineteenth National Computer Security Conference*, pages 353–365, Baltimore, Maryland, 22-25 October 1997. NIST/NCSC.
20. J.H. Saltzer. Protection and the control of information sharing in Multics. *Communications of the ACM*, 17(7):388–402, July 1974.
21. J.H. Saltzer and F. Kaashoek. *Principles of Computer System Design*. Morgan Kauffman, 2009. Chapters 1-6 only. Chapters 7-11 are online: http://ocw.mit.edu/Saltzer-Kaashoek .
22. J.H. Saltzer and M.D. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, September 1975.
23. S. Stolfo, S. Bellovin, S. Hershkop, S. Sinclair, and S. Smith. *Insider Attack and Cyber Security: Beyond the Hacker*. Springer, 2008.
24. K.-P. Yee. *Building Reliable Voting Machine Software*. PhD thesis, University of California, Berkeley, 2007. Technical Report 2007-167; see also Technical Note 2007-136 for the security review; http://pvote.org.
25. L.S. Zegans. The psychology of risks. *Communications of the ACM*, 51(1):152, January 2008. *Inside Risks* column.

Appendix: Some Illustrative Examples of Insider Misuse

The ACM Risks Forum (http://www.risks.org) and an annotated index [16] include many cases of insider misuse. Selected insider cases noted here illustrate a wide range of applications and effects.

• Aldrich Ames (who was in charge of monitoring abuses) was guilty of spying activities inside the CIA. Robert Hanssen had been spying largely undetected for almost 22 years. After the apprehension of Ames, the FBI placed Hanssen in charge of detecting additional spying activity – namely, his own!

• An Autotote ex-programmer with insider knowledge hacked the winning Breeders' Cup Pick-Six horse-race off-track betting system, after a previous trial went undetected. The off-track system transmitted results to a central facility only after the completion of the fourth race. Dummy bets were placed for the first four races with wild cards for every possibility for the subsequent two races. Before transmission, the dummy bets were altered to specify the known winners of the first four races. As a result, one Pick-Six and many Pick-Five choices were winners. Drexel University fraternity buddies were implicated. Programmer Chris Harn was sentenced for only a year and a day in jail, because he helped the authorities incriminate his buddies, who received two- and three-year sentences.

• Harrah's Tahoe was victimized by a $1.7-million payoff scam, with insertion of a Trojan-horse chip and an electronically triggered payoff suspected. An attempted lottery fraud in Pennsylvania resulted when an insider managed to print a winning ticket with a backdated timestamp, but was detected.

• The Washington DC Real Property Tax Administration Adjustments Unit was victimized by a group of insiders who filed for fake property tax refunds for $30 to $50 million – and remained undetected for over 10 years, despite the size of the conspiracy, which included Bank of America employees who knowingly cashed the fraudulent checks. This case may still be in the courts. (Browse on 'Harriette Walters'.)

• Various law-enforcement misuses of databases and thefts of criminal records have been reported, involving police, an FBI employee, and a former Drug Enforcement Agency employee. An IRS agent was accused of giving a defendant tax data on judges and jurors. Social Security Administration employees sold 11,000 Social Security Numbers to activate cards stolen in the mail. Forty people including nine postal workers were arrested in an extensive credit-card fraud in Washington DC. In Virginia Motor Vehicle frauds, illicit driver licenses were sold for as much as $3,500; some years ago, the going rate was only $25. California DMV clerks were fired for similar offenses. Database misuse by 11 prison guards in Brooklyn leaked names of informants to prisoners, warning about impending searches.

• Numerous cases of insider misuse and fraud in banking, credit cards, identity theft, stock trading, security guards, call centers, a hospital nurse changing prescriptions and treatments, ... The recent Madoff Ponzi scheme appears to be just the emerging tip of an insider-fraud iceberg. Volkswagen Corp lost $260 million to a computer-based foreign-exchange fraud; four insiders and one outsider were convicted. A military pay fraud netted $169,000 using a bogus account. A massive New York City tax fraud wiped out $13M in taxes; many insiders were implicated. Joseph

Jett created $350-million phantom profits undetected by Kidder Peabody oversight, and received a bonus of $9M. Many bank customers (48,000 at Wachovia, 600,000 at Bank of America, and others at Commerce Bank and PNC Bank of Pittsburgh) were notified that their financial records were potentially compromised by an insider operation. Insider thefts of sensitive information include 6000 AIDS records stolen from a Miami hospital. Misuse of sensitive personal information has led to many security violations and compromises. In addition, various cases of stolen or lost laptops have involved compromise of unauthorized or inappropriately stored insider information.

• Election irregularities. In 1984, David Burnham reported on election fraud in *The New York Times.* In Colorado, absentee ballot fraud was reported going back to 1984. In 1999, 22 people were indicted in Louisiana in a bribery/kickback scheme. In 2009, five insiders in Clay County, Kentucky, were indicted for previous election frauds, including systematically altering ES&S iVotronic electronic ballots after voters had been intentionally misled about the user interface. Other cases of fraud are also suspected.

Insider Threat and Information Security Management

Lizzie Coles-Kemp and Marianthi Theoharidou

Abstract The notion of insider has multiple facets. An organization needs to identify which ones to respond to. The selection, implementetion and maintenance of information security countermeasures requires a complex combination of organisational policies, functions and processes, which form Information Security Management.This chapter examines the role of current information security management practices in addressing the insider threat. Most approaches focus on frameworks for regulating insider behaviour and do not allow for the various cultural responses to the regulatory and compliance framework. Such responses are not only determined by enforcement of policies and awareness programs, but also by various psychological and organisational factors at an individual or group level. Crime theories offer techniques that focus on such cultural responses and can be used to enhance the information security management design. The chapter examines the applicability of several crime theories and concludes that they can contribute in providing additional controls and redesign of information security management processes better suited to responding to the insider threat.

1 Introduction

This chapter examines the role that traditional information security management approaches play in the defence against the insider threat and the possible enhancements that crime theories, such as Situational Crime Prevention theory, may offer. Information security management is a complex combination of organisational poli-

Lizzie Coles-Kemp
Information Security Group, Royal Holloway, United Kingdom e-mail: Lizzie.Coles-Kemp@rhul.ac.uk

Marianthi Theoharidou
Dept. of Informatics, Athens University of Economics and Business, Greece e-mail: mtheohar@aueb.gr

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*

DOI 10.1007/978-1-4419-7133-3_3,

cies, functions and processes that are used to determine, implement, maintain and update information security countermeasures. The goal of information security management is to enable an organisation to respond to the context in which information processing and storage take place, so that the processing achieves its purpose without the information being placed at undue risk. Information security management provides the processes for evaluating levels of information risk and responding to those levels of risk. As part of this evaluation, the attributes of the different actors involved in information processing and storage are considered. One set of attributes relates to the degree to which the actor is considered to be internal or external to an organisation and in this chapter we refer to this as the property of "insiderness".

The chapter considers a range of definitions related to the insiderness of actors and reflects on the different interpretations of the concept. Organisations rarely share one view of insiderness [16] and information security management approaches have to respond to a range of interpretations of this concept within one organisation. This chapter outlines the components of information security management and evaluates the different ways in which best practice and standards enable an effective information security management response to the risks presented by insiders. Traditionally information security management approaches are focused on compliance and regulation. These approaches do not typically provide the mechanisms for understanding and influencing the different cultural responses to the compliance and regulatory framework. Cultural variations are often found within an organisation and determine different types of behaviour of individual employees or within organisational units. Understanding and influencing organisational culture is a key response to the insider threat and crime theories are explored later in this chapter to see how they can contribute to the development of an information security management approach that both regulates information security and engages with the organisational responses to the regulation.

The chapter concludes with the practical implications of these theories in information security management, which can be used to extend existing information security management methods and practices in order to improve insider risk mitigation.

2 Definitions of Insider and the Relevance to Information Security Management

The concept of the insider is central to many information security management approaches because the defence architecture common for many information security designs is based on the notion of the perimeter (or the absence of it). In this design, actors are, in part, evaluated on which side of the perimeter they reside. In this type of architecture, insiders and outsiders are assumed to have different motivations for access to information and, as a result, are managed in different ways. The concept of insiderness is entwined with notions of trust, homogeneous values, authorisation, empowerment and control. It is a fluid, complex concept and within an organisation

there is a range of perspectives on insiderness in operation at any one time. The degree to which an insider is trusted, the range of actions that the insider is authorised to undertake and the degree to which insiders are considered to have homogenous values significantly impacts not only the types of controls that are used to regulate the insider, but also the extent to which internal controls are regarded as necessary. Different organisational units will typically evaluate the insider risk differently depending on their perspective of the insider.

The broad range of interpretations of the insiderness concept is reflected in the various definitions that are used in information security literature. A standard view of insider is reflected in the following definition:

> "any authorised user who performs unauthorised actions that result in loss of control of computational resources." [3]

Such emphasis on authorisation is common in many approaches to managing the insider threat. In this definition there is a separation between the actor who is authorised and the action which is unauthorised. In an information security management architecture the authorisation of actors is controlled through access management and authentication processes. The separation between authorised and unauthorised actions and the manner in which the separation is enforced depends on organisational culture and structure.

Some definitions found in the literature also give focus to malevolence, where non-malicious (*i.e.*, accidental and non-adversarial) actions are not taken into account:

> "A malicious insider is a current or former employee, contractor, or business partner who has or had authorized access to an organization's network, system, or data and intentionally exceeded or misused that access in a manner that negatively affected the confidentiality, integrity, or availability of the organization's information or information systems." [8]

Using this definition information security management approaches need to take motivation as well as actions into account and this requires adjustments to risk assessment, audit and training and awareness methodologies in order to explicitly include motivation as a factor to be assessed and monitored.

In a case study of a military context, non-human actors are introduced into an insider definition:

> "A current or former human or non-human actor who intentionally exceeded or misused an authorised level of access to CIS, networks, systems, services, resources or data in a manner that targeted a specific human or non human actor or who affected the confidentiality, integrity or availability of the nation's data, systems and/or daily operations." [33]

For some organisations, non-human actors are as important as human actors and are regarded as being capable of performing insider attacks. An example of this is the deployment of mobile ad-hoc networks in military theatres of operation [5] where responses to malicious nodes are considered in the context of key management and operational management processes differentiate between malicious and benign nodal behaviour.

The insider workshop that took place at Dagstuhl in 2008, from which this publication emerged, developed the following definition for the term:

> "An insider is a person that has been legitimately empowered with the right to access, represent, or decide about one or more assets of the organization's structure." [7]

The focus in this definition on the notion of empowerment indicates an authorised individual who both has authority and is implicitly trusted. The degree of trust determines the extent to which an information security management approach deploys monitoring and surveillance as a form of control.

Each of these definitions shows different emphasis on aspects of insiderness. The range of definitions was also reflected when information security managers were asked for interpretations of the term insider, as part of observational fieldwork during the writing of this chapter. An information security manager has to deploy controls across an organisation that reflect the varying levels of trust of insiders found within an organisation. The controls also have to respond to a range of cultural norms and values and the differing perspectives on the extent to which insiders should be controlled. The spectrum of insiderness found in the literature is re-enforced in the definition given by the Centre for the Protection of National Infrastructure (CPNI). This description of insiders highlights the range of types of individuals that can be included in this classification and the diversity of motivation that results in an insider exploiting their authorised access:

> "Insiders can take a variety of forms including disaffected staff, single-issue groups (such as animal rights activists), journalists, commercial competitors, terrorists or hostile intelligence service agents. Their motivations are similarly varied and can range from political or religious ideologies to revenge, status, financial gain and coercion." [9]

This range of definitions shows that there are a number of attributes that an insider may have. Thus, when assessing and responding to information security risk, these attributes must be taken into account. The attributes which we consider relevant to the information security management of the insider are: logical or physical location, authorisation, expected behaviour, motivation and trust. By definition, an insider must have crossed some kind of perimeter (logical or physical) and, by virtue of being within the perimeter, the insider has knowledge of the internal environment. If the insider is authorised to be within that perimeter, then there is a likelihood that further authorised access to information has been granted and that the individual is expected, and possibly trusted, to behave in a certain way. Motivation is critical in determining whether the individual chooses to comply with the expected behaviour.

The interpretation of insiderness is cultural and this means that in order to provide a response to the insider threat, an information security management approach must be able to influence the cultural responses to its regulatory framework. Aspects of organisational culture that need to be influenced are: security culture, transparency of policy, security communication and ownership [9]. These cultural facets are reflected in risk assessment methods designed to consider the risk from insiders [11] but also need to be taken into account in the other information security management processes of audit, training and awareness, review and incident management.

3 Risk and Insiderness

Risk is an integral part of information security management. Information security management can be defined as:

> "A subset of information security which acts as an interface between the organisation and the information security mechanisms, where the interface specifies how, where and which mechanisms are implemented and ensures that they are appropriately maintained." [14]

The concept of information security management is instantiated in frameworks, organisational and technical architectures, and organisational processes. One instantiation that combines all these different elements is the information security management system (ISMS). In the international standard for security management, ISO 27001, this system is based on the concept of the continuous process cycle which ensures that information security controls are:

> "established, implemented, monitored, reviewed and improved, where necessary, to ensure that the specific security and business objectives of the organisation are met." [22]

This continuous process can be depicted as a four stage process (see Fig. 1). Context is described in this diagram as "information security requirements and expectations" and can include interpretations of insiderness. As an interface between the environment in which information is processed and the security mechanisms that protect the information, information security management needs to respond to both the context external to an organisation and the one that is internal to it. In much of the information security management literature, the response to the internal and external context is regarded as an aspect of risk management. Information security regards risk as being composed of: threats, vulnerabilities and assets. Information security risk is calculated as the potential impact of a threat exploiting a vulnerability in the protection of an asset multiplied by the likelihood of this event

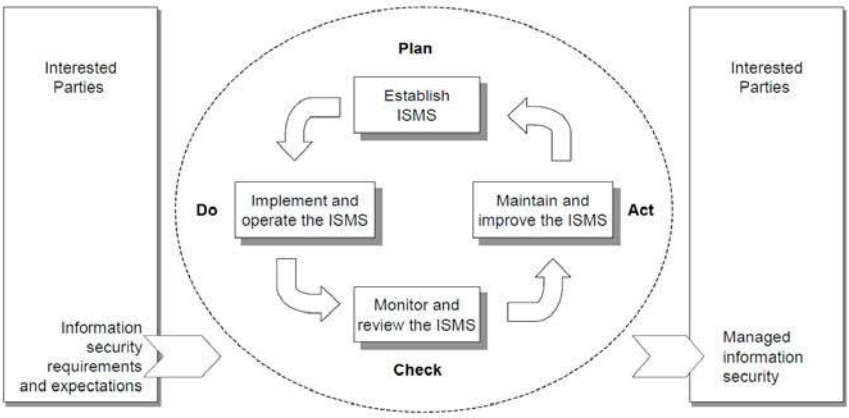

Fig. 1 The Plan-Do-Check-Act cycle [22]

taking place. Insiderness adjusts the information security risk in ways outlined in the remainder of this section. The adjustments caused by insiderness are significant for information security management because approaches have to be designed and implemented that respond to these variations.

From the perspective of information security risk, insiderness potentially adjusts the vulnerability of the information asset, the likelihood of the vulnerability being exploited or the threat being actualised. It could also be argued that insiderness potentially increases the impact level. In the case of likelihood of malicious activities, it could be said to decrease because insiders are often positively motivated towards an organisation, and the likelihood of accidental misuse decreases due to training and education. However, this rule is not absolute and often the likelihood of an insider attack is viewed to increase at periods of organisational change or during times of organisational adversity [11]. Similarly, the vulnerability of many of the information security controls is often considered to increase because an insider is aware of the limitations of the controls, the weaknesses of the organisational processes that maintain the controls and, in particular, how to reduce the likelihood of control misuse being detected. Given that insiders are typically less constrained in their operation of an information system than an outsider, the potential for the extent of damage by an insider is greater. The lack of constraints reflects the empowerment, authorisation and trust that an insider might receive, as well as more detailed knowledge of the information security controls. The lack of constraints is regarded by many organisations as an operational necessity when the level of risk from insiders is considered to be low and the cost of constraining insiders is considered to be high.

The measure of risk to information assets can be further increased by insiders and outsiders combining, so that external motivation and resource combines with internal knowledge and access. CPNI terms the co-operation of internal and external actors as "recruitment" [9] and identifies it as a major factor in increasing the complexity of the risk from insiders, the difficulty of mounting a response and the impact of an incident.

As explained in the previous section, the extent to which insiderness adjusts the risk varies across an organisation and, as a result, information security management requires a range of responses to the insider risk. In addition to countermeasures and security mechanisms, an information security management approach uses training and awareness, education and audit processes as part of the risk response. Training and awareness and education are used to not only disseminate policy but also to develop a security culture which makes attacks from insiders less likely. Feedback from the education process can also be used to check both for policy understanding as well as policy suitability. Audit is a process used to check compliance but can also be used as a check of policy understanding and suitability. It can therefore be seen that information security management processes are used to test defences and influence organisational culture.

Given that the interpretation of insiderness is determined by organisational culture, it is also important to consider the cultural factors that affect the insider risk.

3.1 The Importance of Organisational Culture and the Significance of Cultural Risks

Organisational culture, which comprises an information security culture, has been identified as an important aspect of the information security management response [26, 32]. Schlienger and Teufel describe corporate culture as "a collective phenomenon that is growing and changing over time and can be influenced or even designed by the management of the organisation" [26]. Organisational culture contains a security element and security culture is regarded by many both as an important method of responding to the insider threat and as one of the main factors which affect the extent to which the organisation is at risk from insiders [9]. In the last ten years a stream of security management research [24, 26] has focused on the strengthening of the regulatory technologies as a means of influencing and managing information security culture.

Cultural risks are related to the degree of organisational loyalty, integration with organisational culture and the ease with which misuse can be identified [9]. These factors can manifest themselves as facets of vulnerability that need to be included in a risk assessment that is sensitive to cultural as well as technological risks. Cultural risks will vary across an organisation and it is rare that one completely homogeneous security culture can be established. Therefore, differing cultures need to be assessed as part of the audit process and training and awareness programmes need to be attuned to the variations in perspective on the value of information.

Responding to the variations in risk caused by insiderness and adjusting responses across the different organisational cultures is the role of the ISMS. The following section outlines the structure of the ISMS and evaluates the traditional ISMS response to insiderness.

3.2 Fieldwork on Culture and the Insider Threat

In order to further understand the importance of culture in the management of the insider threat, a team of four practitioners who specialise in the information security management were asked the following three questions as part of a short field study:

1. What are the different aspects of information security culture that you encounter?
2. In what ways is information security culture reflected in the incidents that you encounter?
3. How are security cultures developed in the organisations that you engage with?

The responses and our conclusions are outlined in the subsections below.

3.2.1 What are the different aspects of information security culture that you encounter?

The responses to the question about the aspects of information security culture indicated that there is a number of dimensions to the way in which security culture manifests itself. Namely: perception of the role of insider and the degree to which insiders can be trusted, the gap between perception and practice and the transfer of cultural values from one organisation to another.

These responses indicate that there are a number of cultural aspects that an ISMS needs to identify, understand and influence.

3.2.2 How is information security culture reflected in the incidents that you encounter?

The responses to this question elicited the observation that there is a clear link between the type of security culture and the behaviour of insiders. A further observation was made that the nature of the security culture determined whether anomalies are regarded as incidents or events. Thirdly, it was highlighted that once a security culture degrades significantly, it is expensive to re-define it.

These responses indicate that an ISMS needs to influence culture if it is to adequately respond to the insider threat. It also indicates that the costs of pro-actively influencing security culture needs to be compared with the costs of re-actively correcting a security culture which encourages insider misuse.

3.2.3 How are security cultures developed in the organisations that you engage with?

All responses discussed the iterative nature of culture and how the culture for the organisation as a whole, or ”macro culture” as one respondent termed it, is re-interpreted at least three times before it is adopted by the organisational unit. The conclusions from the respondents reflect the static nature of “macro culture” and the dynamic nature of the cultures (or ”micro cultures”) closest to the organisational unit. An observation was made that the macro culture was most closely linked to the regulatory framework but the micro cultures were more closely linked to the individual and local contexts. Respondents highlighted that individuals, team cultures, introduction of new technologies and strategic imperative, as interpreted at the organisational unit and individual levels, dynamically affect the security cultures at operational tiers of an organisation.

These responses emphasise the occurrence of multiple security cultures within an organisation and indicate how regulatory frameworks have most effect on security culture at the macro level. Organisational units and individuals require additional tools to interpret security culture as it cascades through an organisation.

3.2.4 Responses

The remainder of this chapter responds to these field observations. Section 4 shows how the regulatory structure of an ISMS is adjusted to better support security culture. Section 5 shows how crime theories enhance an ISMS to influence the cultural responses to the regulatory structure.

4 The Structure of the ISMS and Traditional Information Security Management Responses to Insiderness

The success of information security management to respond to insider risks is highly dependent on the ability to have a constant 360 degree view of the organisation. The organisational processes that support the detective mechanisms need to work continuously in order to ensure that the information security mechanisms remain appropriate for the risks that the organisation faces [25]. The functions used to provide this complete view are described in Figure 2. An ISMS's organisational functions and processes are used to identify, implement, monitor and update information security controls. In Figure 2, these are described as the technology where this term is used to refer to the policies, the procedures and the mechanisms which combine to form information security controls.

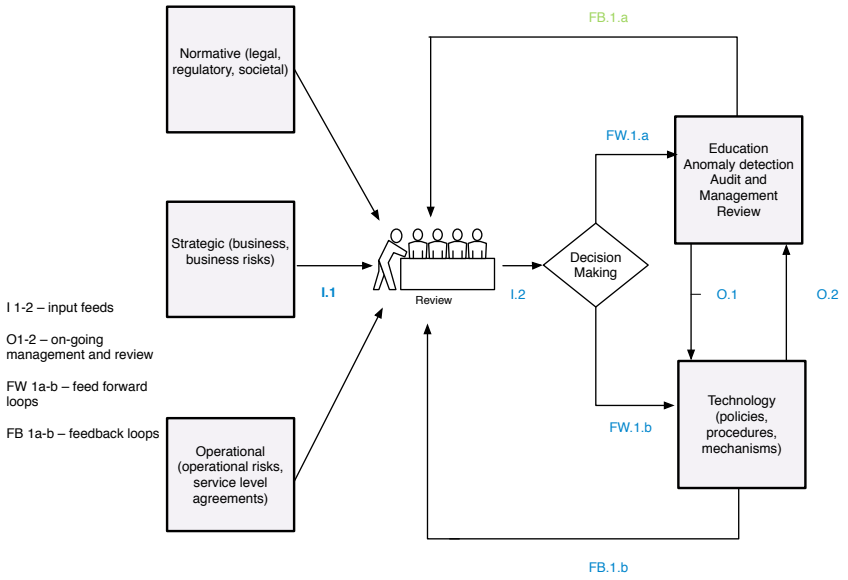

Fig. 2 ISMS and the Continuous Process

As Figure 2 illustrates, an ISMS is composed of a number of organisational functions and processes that interact in order to respond to changes in organisational contexts. The response to change is in the form of both changes to the technology that is deployed and changes to the organisational processes used to implement, monitor and maintain the technology. The traditional view of the ISMS is a regulatory one. The configuration of the ISMS components is determined by organisational state, structure and culture [14, 15] which are factors that will affect both how an organisation determines and responds to any threats from insiders. As explained in the previous section, these factors will also influence the level of insider risk that an organisation faces.

The decision making processes within an ISMS are primarily the risk processes. The role of the decision making processes is to decide which information security controls should be implemented in response to the level of information security risk that an organisation faces. There are three implementation processes used to deploy and monitor the security control decisions. These processes are: audit/review, training and awareness and anomaly detection. Training and awareness is used to communicate information security policies. The audit process is used to assess compliance with information security policy. The implementation processes all feed into each other and also feed back into the risk processes in order to provide the complete view that an ISMS requires.

All regulatory frameworks have operational processes which determine how the framework is responded to. The type of organisational process deployed is related to the culture, in terms of the individual attitudes of the workers and therefore the responses of the operational processes will fluctuate depending on the individuals that are part of each organisational unit. In particular the nature of participation must be considered [19]. It is therefore important to conclude that both the regulatory structure and culture play an important part in the effective management of the insider risk.

4.1 Analysis - Turning an ISMS Inwards

An ISMS needs to point outwards to defend against the external threat and, at the same time, point inwards in order to respond to the insider threat. This dual view forms a full 360 degree perspective of an ISMS. Alterations need to be made to the information security management processes in order to be effective against the insider threat. Configuring the implementation processes in order to respond to risks with a strong element of insiderness, requires the deployment of methodologies attuned to analysing and changing security culture. When responding to the insider threat both compliance and culture are important. An ISMS regulates information security and monitors compliance through its use of policies. However, the implementation processes can also play a role in developing, maintaining and monitoring security culture. Audit and training and awareness can be used not only to train members of the organisation in policy compliance and to assess levels of policy

compliance but can also be used to articulate cultural values, educate users in the philosophy of the security policies and to identify differences in security cultures across an organisation.

The implementation process that is particularly affected by pointing an ISMS inwards is incident management. Differentiating between benign and malicious behaviour is an important aspect of monitoring security culture. As stated in Section 2, expected and trusted behaviour is an attribute of insiderness. The role of responding to the internal context and distinguishing between benign and malicious behaviour of trusted individuals falls to a set of information security controls that can be categorised as detective [33] and the information security management processes and organisational functions that support those detective controls. Walker outlines how both technologies and organisational processes have to be adapted to monitor this aspect of security culture [33]. He emphasises that once the monitoring and detective tools point inwards and concentrate on insider behaviour, it is necessary not only to have a reactive capability but to be able to assess the risk of malicious behaviour and have a proactive capability which contains and prevents malicious behaviour.

As Walker's example of incident management process design highlights, when information security management processes and methodologies are used to respond to insider threats, modifications and enhancements to the design of both components are needed. The breadth of insiderness attributes and the challenges it poses for traditional information security defences place a much greater burden on the detective controls and associated processes.

Viewing risks from an insider perspective also requires a different understanding of the vulnerability and an evaluation of the influence of the security culture. In order to gain this understanding the feedback from the monitoring processes is essential for the decision making processes. This feedback increases both in terms of output and in terms of quality, the more embedded or operationalised an ISMS becomes.

4.2 The Role of Operationalisation

Once an ISMS is designed and initially implemented, the process of operationalisation starts so that an ISMS becomes part of routine organisational unit activity. Operationalisation results in an ISMS embedded within an organisational unit and its operational processes. The activities undertaken to operationalise an ISMS include: training and awareness, education, communication of policy, audit and compliance checking, definition of operational roles and responsibilities, etc. These are also essential activities in the establishment of a security culture. When an ISMS is embedded into an organisational unit there is a direct interaction between the ISMS technologies (policy, processes, countermeasures and procedures) and the cultural responses to the technology. Differences between the cultures of the different organisational units result in different interpretations of policy compliance, different perspectives on information security risk and different emphasis on control. In order to operationalise the ISMS, there need to be many more review fora within an

organisation in order to enable the different parts of an organisation to reflect on the requirements of policy and determine how the policy will be adopted within each part [14]. The development of review fora is one of the main ways in which different parts of an organisation determine its response to the regulatory structures. Review fora can be separated into fora that operate at a governance level within an organisation, at a management level and at an individual level within an organisation. Ultimately ISMS embedding is completed at the individual level, rather than at the level of the organisational unit. The process of embedding is reflected in personnel risk assessment which breakdown personnel risk into organisational risks, group risks and individual risks [11] and enables a security manager to consider the different security cultures that constitute an organisation. These three tiers are also reflected in Schlienger's work when he discusses the operationalisation of security policies. [26]

The design of information security management approaches is largely based on standards and best practice. Perhaps the best known standard is the family of documents that belong to the ISO 27000 series. However, as we discuss in Section 5, these are not the only guidelines.

5 Information Security Management Standards, Best Practice and the Insider Threat

Information security management is the subject of many best practice guides, regulation specific for different sectors of business, legislation and international standards. The vast majority of these approaches focus on regulation and in doing so, address a number of prime insider threats including fraud and theft. There is an emphasis on setting the appropriate security culture from the top of the organisation, and indeed in the informal field observations confirmed senior management attitude as a significant factor in increasing or decreasing insider risk. There are risk methodologies that profile attackers and their motivations but interestingly these methodologies are not included in many of the mainstream information security management standards and best practice guides. This section considers the ISO 27000 family of security management standards and the specific guidance available for managing the insider risk.

5.1 General Security Management Standards

There are a number of standards which are used to design and implement information security management controls and processes. The majority of these standards are control-focused and concentrate on responses to particular types of information security risk. The family of standards which underpins information security management is the ISO 27000 family. The two main standards are ISO 27001 which

presents the ISMS and ISO 27002 which presents the control set used by the ISMS to respond to context [22, 23]. The control set breaks down into twelve areas each of which are characterised in terms of the dimension of information security that they relate to. There are various controls that can be used to respond to the risk from insiders [21]. Table 1 presents the control classifications defined in Annex A of ISO 27001:2005.

As Humphreys discusses, all ISO 27002 control areas have relevance for responding to the insider threat [21]. Broadly speaking, three distinct categories of controls can be identified: controls used to identify insiders from outsiders, controls used to identify unexpected insider behaviour and controls used to influence the development of an organisation's security culture. The majority of the controls in this final category can be found in the set of controls termed "Human Resources Security", which are guidelines to be followed upon recruitment and prior to or post employment. These include, amongst others, personnel screening, disciplinary processes, awareness programs, incident reporting and response. In this category, emphasis is also placed on security policy, awareness programmes and security education. Access control and authentication methods, both physical and logical, are the main control groups used to differentiate between insiders and outsiders (*e.g.*, segregation of duties, controls for advanced users or for specific technologies, *i.e.*, mobile devices). This differentiation is also partly carried out using controls that relate to asset management and information classification, labelling and handling.

The event monitoring, compliance and information security incident management categories are the main control groups for determining unexpected insider behaviour. Finally, the standards include controls for continuity management to minimise the impact of the insider threat. Business continuity and resilience planning is an important response for risks which are either difficult to analyse, complicated to respond to or where the risks are unknown. Insider risks can often be categorised in this way, and therefore a business continuity framework and controls that provide resilience offer a way of reducing the impact of an attack from an insider and reduce the need to define insiderness.

5.2 Guidelines Focused on the Management of the Insider Threat

Similar guidelines to the ones found in ISO 27002 are also included in the 16 techniques suggested by the CERT's guide for insider threat prevention and detection, as found by examining 150 cases of insider incidents that were detected and reported [8]. The controls are not general but are specifically designed for insider threat prevention and detection. These include access control, logging and audit, personnel measures equivalent to the ones of ISO27002, physical and environmental controls, controls for software development, change management, policies, awareness and training programs, backup and recovery and incident response. The 16 proposed practices, their relevance to ISO27002 and their goals are presented in Table 2.

Table 1 Control Domains as specified in Annex A of ISO 27001

	Control Domain	Description	Relevance to the Insider Threat
1	Security Policy	A document which outlines the requirements for security, approved by management and reviewed by the organisation	Articulates the information security stance and the values of the organisation which sets the tone of an organisation's security culture
2	Organisation of Information Security	Structure of roles and responsibilities to implement the policy. The controls apply both to organisations and their third parties.	The initial step in the process of operationalising the ISMS.
3	Asset Management	Identification and management of assets	A point at which risks from insiders to information can be identified.
4	Human Resources Security	Controls for the evaluation of all parties that are involved in the processing of an organisation's information.	A critical section which enables an organisation to identify the degree of insiderness relevant to different groups. It also specifies methods for establishing a security culture and ensuring that individuals understand the requirements of that culture.
5	Physical & Environmental Security	Controls for safeguarding the physical perimeters and for management of information process equipment.	A section that enables the geographical establishment of inside and outside.
6	Communications & Operations Management	Controls related to the management of information processing. They vary, including operational aspects, system maintenance procedures, monitoring, network management and regulation of information exchange.	These controls are a combination of methods used to identify unexpected behaviour as well as establish a baseline of expected behaviour.
7	Access Control	Regulation of access to information. The controls are varied and reflect the many different types of access that is possible. This section also addresses the issue of user management.	They regulate and differentiate between internal and external access to information. The controls related to the user management processes are a means of regulating who is regarded as inside and who as outside.
8	Information Systems Acquisition, Development & Maintenance	Identification and implementation of controls as part of the system design lifecycle. They range from processing controls, to cryptographic controls and the on-going management of technical vulnerabilities.	Can be used to determine the mechanisms for differentiating between insiders and outsiders at a system level.
9	Information Security Incident Management	Controls related to the implementation of an incident management process.	The processes for identifying unexpected behaviour and the responses to unexpected behaviour.
10	Business Continuity Management	The structure of the business continuity framework and the design of the business continuity lifecycle.	These controls provide resilience to insider attacks.
11	Compliance	The requirements to audit both at the organisational and technological levels of an ISMS.	The processes for identifying unexpected behaviour and reporting changes to security culture.

Table 2 CERT's guidelines for insider threat management

Practice / Guideline	ISO Control	Goal
1 Clearly document and consistently enforce policies and controls. 2 Consider threats from insiders and business partners in enterprise-wide risk assessments.	Security Policy	Guidelines that articulate the information security stance and the values of the organisation. They also ensure that the expected behaviour is communicated.
3 Anticipate and manage negative workplace issues. 4 Monitor and respond to suspicious or disruptive behaviour, beginning with the hiring process. 5 Log, monitor, and audit employee online actions. 6 Deactivate computer access following termination. 7 Institute periodic, security awareness training for all employees.	Human Resources Security	These guidelines enable an organisation to identify the degree of insiderness relevant to different groups. They specify a number of methods for establishing a security culture and ensuring that individuals understand the baseline of expected behaviour. They also set the requirements for monitoring and screening for insider threat detection.
8 Track and secure the physical environment.	Physical & Environmental Security	Guidelines that enable the geographical establishment of inside and outside.
9 Use layered defence against remote attacks.	Communications & Operations Management	Guidelines that ensure that the distinction between outside and inside is not dyadic but it is gradual.
10 Implement strict password and account management policies and practices. 11 Use extra caution with system administrators and technical or privileged users. 12 Enforce separation of duties and least privilege.	Access Control	These guidelines regulate in what way the inside-outside distinction is implemented. They also introduce additional restrictions in order to control unexpected behaviour within the organisation.
13 Consider insider threats in the software development life cycle. 14 Implement system change controls.	Information Systems Acquisition, Development & Maintenance	They introduce additional control mechanisms for insider behaviour.
15 Develop an insider incident response plan.	Information Security Incident Management	It defines the process of detecting unexpected behaviour and defining the adequate response.
16 Implement secure backup and recovery processes.	Business Continuity Management	It ensures resilience to insider attacks.

A guide for personnel security from UK's Centre for the Protection of National Infrastructure [9], describes measures very similar to the above. However, it focuses more on the insider's motives and it includes a broader range of organisationally oriented techniques for developing a security culture, which include achieving openness, transparency and communication or applying techniques, such as staff opinion surveys or rewarding procedures for employees. It also provides more detailed guidelines for employee screening upon hiring [10] or dismissal, incident reporting and contracting.

5.3 Analysis of the Contribution of Best Practice and Guidelines

The focus of ISO 27001 is on regulation of information security and the development and monitoring of an information security culture is more of a by-product of this focus. The control set described in Annex A of ISO 27001 provides a generic set of security controls that can be also used effectively for the management of the insider risk. However, the degree of effectiveness depends on the methods used to deploy the controls and the degree to which they are adapted to respond to insiderness. Methods are described in ISO 27002, but these are primarily focused on defending the organisation from the outsider and on regulation and compliance of the internal culture. They do not articulate the specific methods needed when turning the ISMS processes inwards (as described in Section 5.1) and neither do they explicitly focus on the techniques needed to influence cultural responses to the controls framework. The assumption is that by implementing a regulatory and compliance framework, the information security culture will develop. However, as described in CPNI's and CERT's literature, there are additional actions needed in order to understand and influence the cultural responses to this framework.

Some methods are identified in the insider threat management literature, but there are significant gaps in the methods needed to influence, monitor and maintain a security culture. According to CERT's guide, training programs should aim to create "a culture of security appropriate for the organization and include all personnel" [8]. The issue is dealt with solely as a matter of enforcement of values and rules; there are no guidelines that specify how to identify differences in cultural traits within the organisation, how to monitor changes in culture on an individual or group basis or how to achieve the former goal of instilling the expected behaviour in all employees. The CPNI's personnel guide [9] refers more implicitly to security culture within an organisation, as it recognises differences between the existing culture and the desired one. It provides some guidance on how to inspire the desired values in the employees. The guide outlines some incentive mechanisms and insight as to how to monitor changes in behaviour and culture. Such incentives include employee monitoring and staff opinion surveys. It also refers to disciplinary processes and explains how these might affect behaviour within groups. All these recommendations are presented in the form of practical assistance in influencing organisational culture. However the process for understanding how insider behaviour is developed and

how differences in behaviour relate to the different cultural values within an organisation is not explicitly discussed. As discussed in Section 4.1, the configuration of processes is as important as the specification of the controls.

There is also an assumption within both general security management literature and within the insider threat management literature that there is one security culture and that strategic views on security are set from the top of the management hierarchy. This view conflicts with the conclusions from 36 security management case studies [14] and also conflicts with the views of security management researchers [17, 26] and organisational cybernetics writing [6, 27]. Multiple security cultures and the ability of an organisation to set security strategy bottom up as well top down within an organisational hierarchy, requires a more sophisticated set of methods for the establishment and maintenance of appropriate security cultures within an organisation. As described in Section 4.2, the manner in which an ISMS is established and embedded into an organisation helps to develop an information security culture, but this is not explained in either type of literature.

Best practice does not address the characteristics necessary in roles used for the effective management of the insider. In order to develop a security culture, the security manager needs a variety of skills in addition to the traditionally recognised technical ones [4]. These skills are used to identify risks to information that arise from the interaction between organisation and technology. The information security manager needs to be able to both respond to organisational change and adversity and bring about cultural change. In the case of the former, the security manager needs to be able to identify the potential impact to security culture of changes such as organisational re-structuring, change in organisational objectives and the impact of changes to operational processes. In the case of the latter, a security manager needs to be able to identify when the information security policies and procedures are inappropriate for a particular organisational culture and adapt the framework to engender a more effective security response.

In order to develop the set of security culture management methods that an organisation needs to understand and influence the cultural responses to its information security controls framework, it is perhaps helpful to consider knowledge in other disciplines. One such discipline is criminology. The following section considers the contribution that crime theories make to the further development of security culture management methods.

6 Crime theories and insider threat

Insider threat can be viewed as a form of delinquent behaviour in the workplace, meaning that it deviates from the expected behaviour, which is the expressed form of the acceptable norms and policies within the organisation. When examining the factors that lead to an insider incident, one needs to analyse and interpret human behaviour and take into account social or psychological attributes that relate to motive or intent. Since this is a human-oriented topic, one could draw upon the fields

of sociology and criminology, where the ISMS techniques or countermeasures find some of their theoretical roots.

Several crime theories have been applied in the information security field [30]. One of the most well-known and prominent theories is the General Deterrence theory (GDT), which suggests that the deterrence of a crime lies on the certainty and severity of sanctions [28]. One application of the theory is the Security Action Cycle [28], which suggests that an ISMS should handle computer abuse in four steps: deterrence, prevention, detection and remedies. The last three steps provide feedback to the deterrence step.

Other theories are Social Bond theory (SBT) and Social Learning theory (SLT). The first one assumes that every person is naturally inclined towards crime, but is deterred when strong social bonds are shared, such as attachment (to family or peers), commitment to socially accepted goals, involvement in conventional activities, and strong belief in social values [20]. The effect of social surroundings is also explored by the Social Learning theory [2, 29], which suggests four factors that determine behaviour: differential association, differential reinforcement/punishment, definition of behaviour and imitation. The Theory of Planned Behaviour (TPB) focuses mainly on the formation of motives, which are shaped by a person's attitude towards behaviour and the subjective norms of social surroundings [1]. An act is committed when there is sufficient motive and when the perceived behavioural control allows it. The latter refers to how a person perceives the difficulty of an act and its own capabilities.

One more recent theory is Situational Crime Prevention (SCP) [12]. This theory is based on the hypothesis that to commit a crime, a person requires both motive and opportunity. It differs from the other crime theories, in the sense that it does not focus solely on motives. It proposes techniques that can be used to affect the environment of an offender and to reduce the available opportunities which are necessary for a crime to take place. It forms a set of controls, which reduce criminal opportunities in a given context. These measures address issues concerning the formation, management or change of the environment in a five-fold manner: (a) making a criminal act appear more difficult, by increasing the effort required, (b) making a criminal act appear more risky, by increasing the possibility of detection, (c) reducing the expected benefits of an act, (d) reducing provocations that may trigger an offender, and (e) removing the excuses a person can make in order to justify the criminal act. The theory has been applied in numerous cases [13], for different types of crimes (*e.g.*, vandalism, robbery, theft, etc.), and in various contexts (*e.g.*, pubs, football fields, stores, houses, etc.).

6.1 Existing Connections between Crime Theories and Information Security Management

Willison suggests that SCP and its core concepts can be applied in the ISMS, providing a theoretical basis for understanding and addressing the issue of computer

misuse within organisations [34, 35]. ISO27002 was found to have a strong theoretical correlation to GDT and a weaker one to SCP [30]. ISO 27002's control set focuses mainly on increasing effort or increasing risk of detection, by the use of physical and procedural controls, access monitoring and auditing. Such measures also alter the perceived behavioural control, as described by the Theory of Planned Bahaviour, which refers to a person's perception regarding whether the obstacles that exist towards a particular behaviour can be overcome. ISO 27002 also suggests some controls for excuse removal, such as security policy, confidentiality agreements and awareness programs. There are only a few references to measures that reduce rewards (*e.g.*, property marking or encryption). SCP's technique of reducing provocation cannot be identified within the standard. Similar results were observed in an analysis of the CERT or the CPNI guides. Most of the controls in the insider threat management literature also focus on increasing effort or increasing risk of detection and they are equivalent to the ones of ISO27002. Both guides include controls that assist in excuse removal, these controls include: policies, awareness and training programs. In CERT's guide, there is a practice for reducing provocation or frustration, which focuses on avoiding disputes that may escalate to future insider incidents, whereas the CPNI guide offers a more detailed list of guidelines in order to achieve openness, transparency and communication. Measures that attempt to reduce rewards relate to backup, recovery and incident response. Any further strong correlations between best practices and the other theories, such as SBT or SLT, have not been identified [30]. Minor associations refer to practices that try to increase the commitment of an employee to the organisation or disciplinary processes that aim to reduce imitation.

7 Implications of Crime Theories for ISMS Design

As a family of theories, crime theories can change the basis of control selection and may also alter the controls themselves, as well as the ISMS processes. Application of these theories provides an example of turning the ISMS inwards (as described by [25]). Turning an ISMS inwards happens at a number of levels within the system: at the organisational level, at the organisational unit or team level and at the individual level. Assessing insider risks and selecting countermeasures is usually done at the organisational level. Group assessments are less common and they focus more on the particular characteristics of a group of employees that may alter the overall risk assessment. Risk could also be assessed on a per individual basis, according to the insider potential and the opportunity level based on role and access of that particular individual. Such an assessment would make sense for employees that have a more priviledged role or present warning signs that require further investigation. However, the process of carrying out individual level risk assessments is resource consuming, and is not yet feasible on a large scale basis, as assessing a person's intent by technical means is a complex task [11]. The following section examines

the potential input that crime theories could bring to each of these three levels in terms of control selection.

7.1 Application of SCP to the ISO Control Domains

Crime theories interperate individual traits that lead to a crime, but the mechanisms they suggest are not individualised, thus the theories contribute more on an organisational or group level. In this section, we focus on SCP and examine how the ISO control set can be modified according to the 25 techniques proposed by the theory. The goal is to present an extended set of insider threat management controls that apply the concepts of SCP. Willison provides an initial classification of the ISO27002 according to the 5 SCP goals [36], as described in Section 6. A more detailed version is presented in [31], which contains adjustments and corrections, and is enriched with controls found in the countermeasure set of the risk analysis and management methodology CRAMM. In Tables 3 and 4, a modified, extended version of these initial attempts is presented. The tables cover all the 25 techniques of SCP with suggested, applicable countermeasures for insider threat, which can be included in the ISMS design. Controls that were missing have been added, analogous controls for this particular context have been found for all 25 techniques, some of which are suggested by the best practices described in Section 5.2. The resulting Tables 3 and 4 were examined to ensure all countermeasures are relevant for insider misuse, in its wider form, which includes anyone that has authorised access, but is not necessarily trusted.

Analysis shows that the concept of reducing provocation is not a group of controls that is typically considered in an ISMS, but Table 3 shows that a few controls can be included in this group. These techniques have been applied in more volatile environments, like bars or football fields, so they require adjustment to respond to a more corporate environment. The same applies for techniques used in controlling substances, which is not a threat strongly associated with computer misuse, except in occasions of vandalism to equipment or premises, or to threats that relate to user errors critical for the normal operation of information systems. These techniques could be applied however as a different form of screening, which seeks to discover not the use of substances, but other potential situations that may alter the individual's usual behaviour or can indicate a higher possibility of insider threat.

The techniques present a practical way to translate the theory of SCP into a set of insider threat controls, which extend the pre-defined domains of the ISO 27002. They are more detailed, insider-specific controls, presented in a lower level of abstraction and enriched by (a) current practices (*e.g.*, anticipate and manage negative workplace issues) and (b) new controls (*e.g.*, recreational activities or breaks to reduce provocation). It is important to note that SCP does not alter the foundations of the framework but introduces some additional controls that focus on responding to insiderness attributes. More importantly, the measures are not categorised on a domain basis, but according to each of the five goals of the theory. The target of SCP

Table 3 Insider Threat Management Controls under the prism of SCP

Technique	Security Controls
1. Increase Effort	
Harden target	Malicious software protection, physical locks and restrains for critical equipment and media, I/O controls, sensitive system isolation
Control access	Physical: Card/token for access, physical locks for doors, reception desk and security guards (at entry), visitor tags/cards Logical: Authentication techniques (Password, smart card, token), intrusion detection systems, strong remote authentication, firewalls
Screen exits	Physical: Security guards and reception desks, visitor tags/cards, accountability for assets that exit the premises Logical: Firewalls
Deflect offenders	Honeypots/honeynets, segregation of duties, personnel screening, Key splitting
Control Tools	Authentication systems, download control and mbile code protection, web access controls, access removal for ex-employees, removal of administrative rights, restricted use of devices (*i.e.*, USB tokens, wireless access), need-to-know access to information
2. Increase Risks	
Extend guardianship	Escorting of visitors, Supervision of staff in secure areas, guardianship of mobile facilities outside offices
Assist natural surveillance	Open plan offices, incident reporting mechanism (*e.g.*, hotline)
Reduce anonymity	ID tags for staff and visitors, audit trails, event logging
Utilize place managers	Management supervision, two person sign-off, monitoring by system administrators
Strengthen formal surveillance	Intrusion detection systems, security guards, CCTV in areas with sensitive equipment or information, alarms (both physical and logical)
3. Reduce Rewards	
Conceal targets	Minimize information about location of critical equipment or offices, conceal use of PCs when travelling, reduce website details, minimize information on login application screens, use of logical decoys, remove any rank or status information on authentication IDs, DMZs
Remove targets	Clear desk policy, workstation Time-out/Password Protected Screen Savers, paper shredders, secure disposal of old PCs and media, regulate use of USB devices or other media, thresholds on access to resources
Identify property	Property marking, digital signatures, copyright protection, data labeling,
Disrupt markets	Intellectual Right Protection, freeware, open source programs
Deny benefits	Encryption, property marking, software dongles, use of multiple hardware or storage media (backup), business continuity planning, insurance, effective/timely incident handling, crisis management
4. Reduce Provocation	
Reduce frustrations and stress	Pleasant working environment, recreational activities, breaks, employee welfare
Avoid disputes	Anticipate and manage negative workplace issues
Reduce emotional arousal	Selection of user-friendly controls, user participation in the risk analysis process
Neutralize peer pressure	Disciplinary processes
Discourage imitation	Rapid repair for web defacement, prompt software patching, enforcing security policy on incidents or disciplinary procedures

Table 4 Insider Threat Management Controls under the prism of SCP (continued)

Technique	Security Controls
5. Remove Excuses	
Set rules	Security policy, Disciplinary procedures, Conflicts of interest guidelines, Confidentiality agreements, Training /Awareness Program, Third-Party Contracts
Post instructions	Email disclaimers, Security Policy, Access labels for critical areas
Alert conscience	Use of messages, *i.e.*, copyright protection, privacy protection etc., Code of Ethics
Assist compliance	Security education for staff, Single sign-on, Point of reference for security issues
Screening *	Psychometric tests or personality questionnaires, Appraisals, Pre-employment checks

* renamed from Control drugs and alcohol

can be considered as a more specialised one: insider threat management as opposed to more general information security management.

The analysis above shows that SCP can have a direct application to the design of an ISMS in terms of control selection. Organisations are most likely to select and prioritise control strategies in the following order: technical, formal, informal [18]. TCP suggests several technical and formal techniques that increase risk of detection or the effort needed, which aim to (a) act as a deterrent (insider threat prevention) and (b) make compliance audit more effective (insider threat detection). This idea has resonance with the General Deterrence theory and Walker's set of proactive and reactive controls [33]. However, SCP also shifts the emphasis to the informal, more social controls as well, as opposed to solely implementing technical or physical countermeasures that increase the effort or the risk of detection. SCP also places significance on techniques that target the psychology of the insider by removing excuses, reducing provocation or rewards. Additional input for controls can be found by examining the other crime theories. Their input is not solely in the planning phase of a security management process, which includes the selection of countermeasures, but it can be relevant to all the stages of the PDCA cycle and at multiple levels of the system: organisational, organisational unit or individual [25]. In the following section we will examine further ISMS extensions offered by other crime theories and whether these are compatible with SCP.

7.2 Implications for ISMS Process Design

When examining the applicability of General Deterrence theory, it can be seen that in many organisations it is difficult to agree on sanctions for policy violations and, equally, it is often difficult to fully audit and detect misuse. There are varying views

over whether the severity of sanctions is effective enough to deter a determined offender. However, developing an audit process that can more accurately differentiate between compliance and non-compliance can potentially act as a useful deterrent. Equally, on a group level, a management review process that can determine the effectiveness of the ISMS controls in a more accurate way, stands a better chance on identifying which organisational units comply with the security controls and which ones differentiate from the norms, in other words, identifying differences in the security culture. This also assists the organisation in maintaining its policies and apply remedies in a unified form, no matter how severe or not they are. Thus, the second concept of the theory, which is certainty of sanctions, is achieved. Remedies affect the whole organisation and not just an individual. If we consider the Security Action Cycle, which is the main application of the theory in the information security management field [28], turning the ISMS inwards, requires a separate Security Action Cycle that targets insider threats. Such a change would affect all ISMS processes and controls and it would require two different action cycles operating simultaneously. A Security Action Cycle could incorporate the controls for insider threat described by SCP (see Section 7.1) into the four processes found within the cycle of deterrence, prevention, detection and remedies.

The social aspect described in Social Bond theory, Social Learning theory and Theory of Planned Behaviour also provides useful input to process design. These theories focus more on the group or individual level and can be applied to the design of more informal controls. More specifically, all theories highlight the notion of beliefs, attitude or definitions of behaviour, and evaluate whether a person perceives an act as positive or negative. These beliefs need to be similar to the beliefs that are formally expressed in the security policy of the organisation and reflect its overall desired security culture. They refer to the individual perception of acceptable behaviour and these theories could provide means to determine perceptions that differ significantly from the security culture of the group or the organisation. In the same vein, SCP suggests controls that aim to minimise such differences, *i.e.*, the remove excuses controls. This set of techniques could be included in the educational processes of the ISMS, in an attempt to unify the security culture of the organisation. They can be applied on a group or organisational level.

The above crime theories also help interpret group security cultures and their variations that may lead to differentiated behaviour within an organisation on an organisational unit basis. In particular, they describe a category of attributes that need to be considered when evaluating the risks from insiders, such as imitation or the effects of association with peers. More specifically, Social Bond theory emphasises the importance of the commitment of a person to the commonly accepted goals, security goals in this case, and Social Learning theory implicitly refers to the effect of imitation and differential association. The Theory of Planned Behaviour names these concepts as subjective norms and also refers to the influence of the environment. This means that an employee's behaviour can be affected positively or negatively by their peers or superiors. Differential reinforcement/punishment (Social Learning theory) refers mainly to whether non-compliance is detected and how it is treated. If non-compliance by peers remains undetected and remedies are not

applied, the Social Learning theory suggests that an individual is more inclined to adopt non-compliant behaviour. Therefore, these theories can contribute to discovering variations from the security goals and they highlight areas that audit processes could focus on. As a follow-on, education and awareness programs can focus on the design of group-based security awareness programs, based on the findings of the ISMS audit that focuses on assessing the impact of the behaviour of associates. The concepts of imitation and peer pressure are also found in SCP theory, with the techniques of remove provocation. The idea is to remove influences of the environment that may affect negatively the individual and audit can be a process that is used to facilitate this environmental change.

In addition, these theories describe a series of individual and organisational states that might warrant inclusion in the audit process. For example, Social Bond theory refers to the bonds of involvement and attachment. An application of these concepts could be the introduction of some form of individual screening that seeks to determine an employee's commitment to the organisation, the level of attachment to peers or the level of attachment to a career. Low social bonds would trigger further audit processes. Individual risk assessments could guide audit processes to identify potential frustrations that may lead to disgruntlement. From the organisational point of view, involvement of the users in the design of the ISMS processes may strengthen the bonds of the employees towards the organisation and assist in their understanding of the relevance of security goals.

All the above deterrent, prevention or detective mechanisms relate to General Deterrence theory and use a corrective action cycle which is also found in the ISMS. Corrective actions can include disciplinary processes or insider incident handling processes. A combination of the remedies concept in General Deterrence theory and SCP would result in enriching the security action cycle by using the techniques that remove benefits (Reduce reward techniques). The goal of enhancing the security action cycle is two-fold: (a) reducing the expected benefit on an individual level, which acts as a deterrent and (b) ensuring resilience and business continuity on an organizational basis, which minimises the insider threat impact.

7.3 Summary of Crime Theory Contribution

As mentioned above, the social theories aim towards interpretation and understanding of an individual's behaviour and can be mainly applied to audit/screening processes in order to provide feedback to the ISMS. General Deterrence theory focuses on how the processes are organised in a Security Action Cycle, whereas SCP aims to modify the environment of the organisation based on the input from the audit and screening processes. The potential changes to an ISMS are summarised in the table below (see Table 5).

Currently, as described in the ISMS literature, the incident management process is based on the PCDA cycle. However, the Security Action Cycle identifies five processes that should be considered when designing the incident management process:

Table 5 Input of crime theories in the ISMS

Theory	Concept/Technique	Relevance to Information Security Management
GDT	Deterrence, Prevention & Detection	New security action cycle that points inwards
SCP	Effort Increase	Processes and controls that inhibit variations from policies
SCP	Risk Increase	Audit or screening processes and controls
GDT	Certainty of sanctions	Audit process to examine compliance
SLT	Differential Association	Process to assess effectiveness of controls
SLT	Differential Punishment	Disciplinary process/ Incident response
SBT	Beliefs	Process to evaluate security culture and discover differences on individual or group level
SLT	Attitude	
TPB	Definitions of Behaviour	
SCP	Removal of Excuses	Process to minimise differences and variations in security culture on individual or group level Training and awareness processes
SBT	Commitment to common goals and the organisation	Process to discover variations from the expected behaviour
TPB	Subjective Norms	User participation in the ISMS Design
SBT	Imitation, Attachment to Peers	Processes and controls that aim to neutralize peer pressure and discourage imitation
SLT	Differential Association	
SLT	Differential Punishment	Processes that identify possible signs or causes for unexpected behaviour
SCP	Removal of Provocation	
SCP	Removal of Benefits	Resilience and business continuity processes

deterrence, prevention, detection, remedies and feedback. These modifications affect one or more of the PDCA steps or, as mentioned above, a new inwards looking PDCA cycle could be defined. As Table 5 indicates, crime theories can affect several ISMS processes of the cycle.

8 Conclusions

There are many definitions and interpretations of an insider and an organisation must respond to this range in its information security management approach. The degree of insiderness and the attributes an organisational unit takes into account, affect the perception of insider risk, the degree of internal control and the types of controls deployed. Information security management encompasses not only compliance and regulation but also the cultural responses to the regulatory and compliance framework. Such responses are not only determined by the enforcement of policies but by various factors that relate to the psychology of the individual or the sociology of the different organisational units. Understanding and influencing cultural responses is an important aspect of responding to the insider threat as it stems from

non-compliance, unexpected behaviour and abuse of organisational trust. However, traditional approaches refer mainly to achieving compliance or enforcing preventive and detective technical controls and less on understanding and influencing culture. As described above, crime theories offer insight into explaining insider behaviour. Consequently, they can provide methods and approaches to information security management design that make it more effective in the management of the insider risk. Crime theories can contribute additional security management techniques and controls, adjustments to the ISMS processes and potentially different action cycles that operate simultaneously on insider and outsider defence.

Acknowledgements Thank you to Terrence Walker, Royal Holloway, for his help with the collection and summarising of the field study results.

References

1. Ajzen, I., Fishbein, M.: Understanding attitudes and predicting social behaviour. Englewood Cliffs, Prentice-Hall, NJ (1980).
2. Akers, R.L.: Deviant behavior: a social learning perspective. Belmont, CA (1977)
3. Anderson, R.H., Bozek, T., Longstaff, T., Meitzler, W., Skroch, M., Van Wyk, K.: Research on Mitigating the Insider Threat to Information Systems - no.2, RAND Conference Proceedings (2000)
4. Ashenden, D.: Information Security management: A human challenge? Information Security Technical Report. **13** (4), 195–201 (2008)
5. Balfe, S., Reidt, S.: Key Deactivation Strategies in MANETs: A Survey (2008) Available online.
   `http://www.sreidt.com/wp-content/uploads/2009/01/reidt2008\textunderscorerevocation.pdfCited20July2009`
6. Beer, S.: The Heart of Enterprise. John Wiley & Sons (1995)
7. Bishop, M., Gollmann, D., Hunker, J., Probst, C.W.: Countering Insider Threats, Dagstuhl Seminar 08302 (2008)
8. Cappelli, D., Moore, A., Trzeciak, R., Shimeall, T.J.: Common Sense Guide to Prevention and Detection of Insider Threats, Ver. 3.1. Carnegie Mellon University (2009)
9. Centre for the Protection of National Infrastructure: Ongoing personnel security - A good practise guide. United Kingdom (2008)
10. Centre for the Protection of National Infrastructure: Pre-Employment Screening - A good practise guide, 3rd Edition. United Kingdom (2009)
11. Centre for the Protection of National Infrastructure: Risk Assessment for Personnel Security - A guide, 3rd Edition, United Kingdom (2009)
12. Clarke, R.: Situational crime prevention: theory and practice. British Journal of Criminology. **20**, 136–137 (1980)
13. Clarke, R.: Situational crime prevention: successful case studies. Harrow and Heston, NY (1997)
14. Coles-Kemp, L.: Anatomy of an Information Security Management System. Ph.D. thesis, King's College, University of London (2008)
15. Coles-Kemp, L.: The Effect of Organisational Structure and Culture on Information Security Risk Processes. Risk Research Symposium 2009 (2009). Available online.
   `http://www.kcl.ac.uk/schools/sspp/geography/research/hrg/papersCited20July2009`

16. Crinson, I.: Assessing the 'insider-outsider threat' duality in the context of the development of public-private partnerships delivering 'choice' in healthcare services: A sociomaterial critique. Information Security Technical Report, **13** (4), 202–206 (2008)
17. Dhillon, G.: Managing Information System Security. Macmillan Press, London (1997)
18. Dhillon, G., Silva, L., Backhouse, J. (2004) Computer Crime at CEFORMA: A Case Study. International Journal of Information Management, **24**, 551–561 (2004)
19. Drenth, P.: Culture Consequences in organizations. In.: Drenth, P.J.D., Koopman, P.L., Wilpert, B. (eds), Organizational Decision-Making under Different Economic and Political Conditions, 199–206 (1996)
20. Hirschi, T.: Causes of delinquency. Berkeley, University of California Press, CA (1969)
21. Humphreys, E.:Information security management standards: Compliance, governance and risk management. Information Security Tech. Report, **13** (4), 247–255 (2008)
22. ISO/IEC 27001:2005, Information technology - Security techniques - Information security management systems - Requirements (2005)
23. ISO/IEC 27002:2005, Information technology - Security techniques - Code of practice for information security management (2005)
24. Martins, A., Elof, J.: Information Security Culture. In: Proc. of IFIP TC11 17th International Conference on Information Security (SEC2002), Cairo, Egypt. IFIP Conference Proceedings **214**, 203–213 (2002)
25. Overill, R.E.: ISMS Insider Intrusion Prevention and Detection. Information Security Technical Report, **13** (4), 216–219 (2008)
26. Schlienger, T., Teufel, S.:Information Security Culture: The Socio-Cultural Dimension in Information Security Management. In: Proc. of IFIP TC11 17th International Conference on Information Security (SEC2002), Cairo, Egypt. IFIP Conference Proceedings **214**, pp. 191-202 (2002)
27. Schwaniger, M.: Managing Complexity - The Path Toward Intelligent Organisations. Systemic Practice and Action Research, **13** (1999)
28. Straub, D.W., Welke, R.J.: Coping with systems risk: security planning models for management decision making. MIS Quarterly, **22** (4) 441–465 (1998)
29. Sutherland, E.: Criminology. J.B. Lippincott, Philadelphia (1924)
30. Theoharidou, M., Kokolakis, S., Karyda, M., Kiountouzis, E.: The insider threat to Information Systems and the effectiveness of ISO 17799. Computers & Security, **24** (6), 472–484 (2005)
31. Theoharidou, M., Gritzalis, D.: Situational Crime Prevention and Insider Threat: Countermeasures and Ethical Considerations. In: Tavani, H. *et al.* (Eds.): Proc. of the 8th International Computer Ethics Conference (CEPE-2009), Greece (2009)
32. von Solms, B.: Information Security - The Third Wave? Computers & Security, **19** (7) 615–620 (2000)
33. Walker, T.: Practical management of malicious insider threat - An enterprise CSIRT perspective. Information Security Technical Report, **13** (4), 225–234 (2008)
34. Willison, R.: Understanding and addressing criminal opportunity: the application of situational crime prevention to IS security. Working Paper Series 100. Dept. of Information Systems, London School of Economics and Political Science (2001)
35. Willison, R.: Understanding the offender/environment dynamic for computer crimes: Assessing the feasibility of applying criminological theory to the IS security context. In: Proc. of the 37th Hawaii International Conference on System Sciences (2004)
36. Willison, R.: Understanding the perpetration of employee computer crime in the organizational context. Working paper no. 4, Copenhagen Business School (2006)
37. Willison, R.: Understanding the perpetration of employee computer crime in the organizational context. Information & Organization, **16** (4), 304–324 (2006)

A State of the Art Survey of Fraud Detection Technology

Ulrich Flegel, Julien Vayssière, and Gunter Bitz

Abstract With the introduction of IT to conduct business we accepted the loss of a human control step. For this reason, the introduction of new IT systems was accompanied by the development of the authorization concept. But since, in reality, there is no such thing as 100 per cent security; auditors are commissioned to examine all transactions for misconduct. Since the data exists in digital form already, it makes sense to use computer-based processes to analyse it. Such processes allow the auditor to carry out extensive checks within an acceptable timeframe and with reasonable effort. Once the algorithm has been defined, it only takes sufficient computing power to evaluate larger quantities of data. This contribution presents the state of the art for IT-based data analysis processes that can be used to identify fraudulent activities.

1 Introduction

Nowadays, it would be impossible to run a modern company without processes supported by IT, and such a company would no longer be competitive due to the enormous costs involved. This was the reason why large companies first converted their financial accounting processes to IT and then followed up with other processes. As a consequence, *Enterprise Ressource Planning* (ERP) systems can today be found in innumerable companies. The implementation of IT resulted in many improvements.

Ulrich Flegel
SAP AG, Vincenz-Prießnitz-Str. 1, 76131 Karlsruhe, Germany, e-mail: ulrich.flegel@sap.com

Julien Vayssière
Smart Services CRC, Australian Technology Park, Locomotive Workshop Suite 9003, 2 Locomotive St., Eveleigh NSW 2015, Australia, e-mail: julienv@smartservicescrc.com.au

Gunter Bitz
SAP AG, Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany, e-mail: gunter.bitz@sap.com

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_4,

Processes became faster while costs dropped. Vast paper archives became a thing of the past, giving way to IT systems that allowed rapid access to data.

At the same time, however, the introduction of IT prompted the loss of a human control step. For example, orders would leave the company electronically and fully automated, with no human intervention. For this reason, the introduction of new IT systems was accompanied by the development of the authorization concept. In practical terms, this was not a new development, but simply a question of adjusting existing IT-based access control principles to suit business processes. The signature concept was replicated in the form of the approval workflow. A system of authorization now made it possible to restrict certain tasks to certain employees in the company, and to share critical business processes between several employees.

The security of the electronic world of business stands or falls with the quality of the *Separation of Duties* (SoD) model.That is why it is of paramount importance that due care is exercised in analysing this system for risks and then minimizing them. This is, of course, also supported by software nowadays (*e.g.*, SAP GRC Access Control [12]).

But since, in reality, there is no such thing as 100 per cent security; auditors are commissioned to examine all transactions for misconduct. Given the sheer quantity of data that is processed today, it would be utterly futile to attempt to check all of it manually. It would take an entire army of auditors to check just one fiscal year's worth of data in a reasonable amount of time. Since the data exists in digital form already, it makes sense to use computer-based processes to analyse it. Such processes allow the auditor to carry out extensive checks within an acceptable timeframe and with reasonable effort. Once the algorithm has been defined, it only takes sufficient computing power to evaluate larger quantities of data.

1.1 Data Analysis Methodology

1.1.1 General

It is not always possible to establish a clear division of functions. Particularly in small and midsize companies, the personnel requirements to support a full SoD implementation would exceed reasonable levels, and would lead to a hike in administrative costs. Furthermore, the SoD implementation would have to take reserves into account to cover illness and vacation. However, since accounting and reporting tasks can be managed by considerably fewer staff than the number of roles involved in a full SoD implementation, different roles must inevitably be covered by the same person.

Generally, people are prepared to accept the higher risk factor that accompanies a partial SoD implementation for the sake of saving on administrative costs. That is why the analysis of data is particularly important in this area in order to discover fraudulent activities in the company.

In large companies also, data analysis has a twofold benefit. On the one hand, it is possible to uncover cases where employees have discovered and exploited gaps in the SoD concept, or where two or more employees have collaborated in order to deliberately bypass SoD. On the other hand, such findings serve as a basis for improving the SoD concept in order to prevent the same activities from recurring in the future. And last but not least, even the best SoD concept will always contain process steps that are to be carried out by one single employee who can find a way to act fraudulently. This is the case, for instance, with credit cards, where a further approval step would reduce the flexibility of the credit card to absurdity.

For all companies, it is therefore worthwhile to regularly analyse all available posting data. It is not of particular significance whether the data is stored in several different systems, since it is normal to work with data exports and imports anyway. In fact, it is usually necessary given that the analysis functionality is usually provided in an external tool.

1.1.2 Procedure

While data analysis is used in different ways, a standard procedure has also become established [5, 6, 7, 10]. Data is analysed in a GUI-based analysis tool (such as ACL [2], IDEA [4]) or in an advanced spreadsheet (*e.g.*, Microsoft Excel or OpenOffice Calc).

Using known fraud and corruption indicators, queries are programmed to test for occurrences of these indicators.

The results then undergo a manual plausibility check before being passed on to the auditors. This takes place because the presence of certain indicators does not constitute conclusive evidence of deliberately fraudulent behavior. Many such results are false-positives, where rules have perhaps been violated due to inadequate internal processes, but the employee did not act with fraudulent intent.

Particularly good analysis algorithms immediately conduct cross reference checks of the indicators found so that the evidence becomes stronger.

In a ”scoring” procedure, points are awarded in relation to the weighting allocated to each indicator, and these points are added up to achieve different views of the data:

- Total for each employee
- Total for each business process
- Total for each project

If one of these points totals exceeds a predefined threshold, the respective employee, business process, or project undergoes a thorough audit.

2 Survey of Technology for Fraud Detection in Practice

In this section we will first have a look at the available general approaches for detecting fraud and then survey the state of the art of software tools used in practice for detecting insider fraud. We will then address the question of why and how fraud detection is different from intrusion detection at host or network level. Finally, we will list a number of challenges faced by designers of fraud detection tools.

2.1 General Approaches for Intrusion and Fraud Detection

The art and science of detecting intrusions and fraud in monitoring and transaction data has a history of over 25 years and a complete survey of the literature in this field is clearly out of the scope of this document. Approaches to intrusion detection can be classified by the kind of behavior that is modeled in order to detect intrusions. For each class a host of diverse methods for implementing the approach have been proposed in the literature, here we will only name a few that are well known. Each approach has inherent advantages and shortcomings. The following classification translates seamlessly to the domain of fraud detection.

Table 1 Desirable properties of intrusion and fraud detection methods; +: the method exhibits the desired property; –: the method does not exhibit the desired property

Desired properties	Misuse Detection	Specification-based Detection	Anomaly Detection
Low false positives, esp. for unknown behavior	+	+	–
Low false negatives, esp. for unknown behavior	–	+	+
High specificity of alarms	+	–	–
Requires no training of models	+	+	–
Requires no manual modeling	–	–	+

Employing models of intrusion behavior for intrusion detection is relatively straightforward. This approach is denoted as **misuse detection**. Known domain knowledge is modeled in a suitable framework, such as finite state machines, Petri nets and regular expressions. The stream of observed and relevant events is then matched against these models. The occurrence of a match triggers an alarm. Alarms may provide attack-specific information for mitigation, since the respective domain knowledge already exists. Since an alarm is only generated, if a model has been matched, by definition no false positives would occur. This assumes that the modeling framework is sufficiently expressive and that the models are sufficiently specific. However, since only already known attacks can be represented, unknown attacks go undetected. A continuous manual and thus costly maintenance of the model knowledge base is necessary.

The approach of **anomaly detection** assumes

1. that normal behavior, be it the behavior of a user, a protocol or process, can be accurately described, *e.g.*, statistically,
2. that attacks deviate from normal behavior, and
3. that all deviations from normal behavior represent attacks.

At design-time the normal behavior is learned from training data, which must not contain any attacks. The machine learning approaches employed to do so are manyfold: statistics, Markov processes, data mining, support vector machines, artificial neural networks, artificial immune systems, etc. During runtime, observed behavior is compared to the learned models of normal behavior and deviations are flagged as alarms. Since the approach considers deviations from normal behavior, no explicit knowledge about attacks is required and unknown attacks can be detected. Since, however, no explicit attack knowledge is used the alarms are unspecific and cannot provide further information on the attack and how to mitigate its effects. Anomaly detection systems usually come with a much higher false alarm rate than misuse detection systems due to the following reasons. The underlying assumptions do not hold entirely:

1. Modeling behavior may not be possible with the required accuracy, the machine learning methods often are inherently fuzzy and the selection of features may be specific to the given environment and may not translate to other environments. The initial training therefore is a costly process that needs to be adapted to the target environment.
2. There is no general argument why attacks would always deviate from normal behavior, and the smaller the deviations the system needs to detect, the more heavily it is prone to false alarms.
3. There is no general argument why all deviations from normal behavior would need to be attacks. Non-attack deviations result in false alarms.

The third approach is called **specification-based detection**, since deviations from an a priori defined specification of security-conforming behavior are detected. At design time a specification of allowed behavior is generated manually or automatically. Automated approaches usually work in domains with strongly limited behavior space, such as network protocols. Manually modeling allowed behavior for users is a cost-intensive task. During run-time observed events are compared to the specification and deviations are flagged as attacks. Implementation techniques leverage compiler technology, where the specification is defined as a grammar and a parser checks the input events against the grammar. Since no explicit attack knowledge is used, alarms are unspecific. However, this approach does not require initial training and comes with low false positive and false negative rates.

Table 1 summarizes the described properties of all three detection approaches.

In the domain of fraud detection, mainly anomaly detection techniques are used to flag unusual behavior that needs further manual examination and misuse detection heuristics are employed to detect known domain-specific fraud schemes.

2.2 State of the Art of Fraud Detection Tools and Techniques

Fraud detection tools work by analysing data already stored and processed by a number of information systems. It is important at this point in the discussion to note that most of the tools made available to fraud auditors do not go beyond importing data and distilling it for the purpose of displaying information from which the fraud auditor can extract cases of suspected fraud. These tools do not usually directly identify potential fraud cases or classify them through any sort of machine-learning technique. The classification stage is entirely left to the human. This of course explains why generic office tools (*e.g.*, Microsoft Access and Excel) are as popular with fraud auditors as specialised packages such as ACL and IDEA.

A useful categorization criterion for tools used by fraud auditors is the degree of coupling between the fraud detection tools and the company and its existing information systems. This yields three categories of fraud detection tools: generic, specialized and custom-built tools.

2.2.1 Generic Tools and Computer-Assisted Audit Tools

These tools provide the user (internal auditor for example) with a toolbox for importing, processing and exporting data for the purpose of fraud detection. However, very little knowledge about what types of fraud to look for and how to detect it is built into these tools. Typical examples would be ACL (Audit Command Language) [2] and IDEA (Interactive Data Extraction and Analysis) [4] but also generic data analysis packages such as SAS or Microsoft Excel. Even if ACL and IDEA are sold as "user friendly", they still require a certain amount of training for fraud auditors to be able to use them. When used to analyse the data extracted from an ERP system, for example, it is not uncommon to find that intimate knowledge about the data structures used in the ERP system is required in order to perform the data analysis effectively.

A study performed by the Institute of Internal Auditors in 2006 to find out which technology auditors rely on revealed that ACL was the clear leader in the field of Computer-Assisted Audit Tools (CAATS). However, this needs to be put in perspective with the fact that, when it comes to data analysis, ACL ranked second behind Microsoft Excel and in front of Microsoft Access. Clearly, CAATS do not deliver sufficient auditing-specific value to justify dropping standard tools altogether.

When asked specifically about fraud detection, users express a preference for CAATS, even though Excel and Access together represent the tool of choice for 40% of the persons interviewed.

One could also mention functionalities embedded in ERP software, such as the *Auditing Information System* (AIS) provided by SAP ERP. It is a set of reports that an auditor can run to generate the information most needed in financial audits and fraud audits. Among those reports are fraud-specific ones such as multiple invoices, one-time vendor accounts and analysis of payment terms.

2.2.2 Specialised Tools for Fraud Detection

We refer here to tools for fraud detection that are specific to a given industry or function. These tools are typically very effective at the analysis of a certain type of fraud but lack the flexibility to adapt to adjacent markets.

An example is the Triversity company that SAP acquired in 2005. Triversity is specialized in delivering software to the retail sector. Triversity tools help managers of retail chains detect fraud committed by store managers, and helps store managers to detect fraud committed by their employees.

The tool can be seen as a set of configurable reports that are run on the data collected from POS (Point of Sale) devices. This data documents everything down to a keystroke on the cash register. A manager may want, for example, to study the ratio of cash purchases over credit card purchases per checkout employee since a ratio lower than the average may be an indication of an employee pocketing cash from cash purchases. Another example would be the rate of product returns across stores since this is an indicator of staff or management creating fake product returns in order to pocket the reimbursed cash.

The insurance business also has a range of players for insurance claim fraud detection. Insurance Services Offices [9] for example, acts as a data hub for a large number of insurance companies for the purpose of detecting fraud among customers. This data can be visualized with powerful specialized tools such as NetMap Analytics [11] that help auditors relate disparate pieces of information such as claims, people, addresses and phone numbers in order to detect scams.

Specialised tools exist also, of course, in the banking sector. Being able to detect credit card fraud is a particularly challenging task since the entire fraud detection step takes place within the few seconds between the card swipe and the decision to authorize or reject a transaction. Detection techniques are a combination of customer scoring and ad-hoc rules learnt from experience about which transactions are likely to be fraudulent[8].

Anti-Money Laundering is another niche market for fraud detection tools. Even if banks do not have a direct monetary incentive to detect dirty money, there are strong regulatory frameworks in place that force them to do so. A number of tools exist and are used by most banks. However, banks do not invest much effort into correlating data between banks in order to detect the more advanced money-laundering schemes. They usually rely on passing information about suspicious transactions on to the banking regulatory authorities who may then investigate the matter on their own.

2.2.3 Custom-built Tools

Custom-built tools form the silent majority of fraud detection tools. They range from crude data processing in Excel to sophisticated Access applications or even custom developments conducted directly inside ERP systems.

The major advantage of custom-built tools is that they are uniquely fitted to the business of the company at hand. Even so, anecdotal evidence reports that it may take a couple of years to get things right. Extracting the information and lowering the rate of false positives generated by the analysis stage are the main challenges.

The predominance of Excel as a platform for custom-built tools is reinforced by the fact that Excel is a popular exportation format for data stored in enterprise applications since the spreadsheet paradigm fits the paradigm of the relational database table quite well.

It is of course difficult to provide a survey of custom-built tools since they are not usually publicized. One should not however underestimate their degree of sophistication: some of them come complete with a workflow component for auditable processing fraud alerts and email or SMS notification of new cases.

Large auditing firms are also known to develop their own tools for fraud detection. However, since the tool captures in software the differentiating advantage of the auditing firm, *i.e.*, the auditor's know-how, these tools are not shared.

There is one example of a custom-built tool that has been publicized: the Sherlock tool developed by the auditing firm *PriceWaterhouseCoopers* (PWC) [3]. This tool takes as input a set of accounting statements produced by a large number of companies. In this set are a set of documents that are known to be fraudulent. The system is trained to classify financial statements as indicative of a company that may be "cooking the books", *i.e.*, doctoring financial statements to their advantage. The objective for a large auditing firm such as PWC is obvious: improving the efficiency and the effectiveness of the audit procedure. The classifier used is a Bayesian classifier together with the expectation-maximization (EM) algorithm. A lot of other approaches for applying machine learning to fraud detection have been proposed in the research literature. Neural networks feature prominently here.

However, even though some results from the research community have managed to make their way into custom-built or specialized software, such as credit card or telecom fraud detection, they have not yet been made available through generic tools such as ACL or IDEA. The reason is most likely that applying a classification technique to a given problem in order to obtain valuable results is no "push button" software, but rather a project in itself which requires both a lot of expertise with classification techniques and solid training data. It is actually no surprise that only an auditing firm such as PWC, with thousands of customers across different industries and millions of accounting documents at hand, could effectively train a system to detect financial statement fraud.

3 Why Fraud Detection is not the Same as Intrusion Detection

A number of research attempts have been made to investigate how techniques and tools developed for the purpose of detecting intrusions on computer network or into computers at the level of the operating system could be re-used for the purpose of detecting fraud.

Even though this may be a good idea in some cases, we believe there are a sufficient number of differences between the two application domains that warrant a specific approach for research in fraud detection. Let us try to list a few:

An Intrusion Detection System (IDS) typically scans logs of network traffic or OS-level events to detect known patterns of attack. These patterns are known either because they have been observed many times in the wild, or because the particular piece of software that implements the attack (the so-called *exploit*) has been reverse-engineered. Once an exploit is released, people with little or no knowledge about the particular vulnerability that is being exploited can perform the attack.

Insider fraud detection, on the other hand, is about rare events. We are not talking about an email gateway being probed thousands of times a day for a vulnerability patched long ago, or about a Web server under attack from a worm such as Code Red. For a medium-sized company, we may be talking about an accountant putting in a fake invoice in March, a warehouse worker driving home with a brand free new TV in September and the head of purchasing favouring a high school friend in a competitive bid in exchange for a ski vacation around Christmas.

Even if insider fraud is a problem for all organisations, it is still a rare event when compared to the day-to-day operations of a company. In addition, each case of fraud is different. This is because most fraudsters are not criminal masterminds that carefully plan fraud schemes for months and exchange information in back alleys with other fraudsters before perpetrating the act. The typical fraudster may be your average employee who, under circumstances of unbearable personal financial stress (gambling, divorce, drug addiction, etc.), and having stumbled upon weaknesses in the internal controls of the company, decides to step over the line and use the trust bestowed on him by his employer for his personal gain. Fraud events are therefore rare and polymorphic, posing challnges for the misuse detection approach.

That being said, it is known to IT security experts that malware and exploit code today is polymorphic and stages multi-step attacks, to the point that naïve misuse detection approaches no longer work. This indeed brings IDS techniques closer to fraud detection. Still, we believe the variability of attacks mounted by insiders is a lot greater than for network attacks. Sadly, there is not enough data available on fraud to allow us to substantiate this claim. Ideally, we would like to study the log files for a fraudster that had been active for several years before being caught[1] to find out if he tried to vary his attacks to avoid detection or, on the other hand, if not being caught made him complacent and he repeated the same fraud pattern over and over again. The lack of fraud audit data makes it difficult to evaluate and compare fraud detection methods.

[1] According to the ACFE Report to the Nation [1] it takes on average 18 months to detect a case of fraud after the fraudster starts perpetrating fraud.

4 Challenges for Fraud Detection in Information Systems

To finish, we provide a non-exhaustive list of the challenges faced by fraud detection tools in information systems:

The move to the business level: Most tools are still meant to be used by a hybrid of internal auditor and ERP administrator. Typically, they are used by tech-savvy auditors. We need tools that speak the language of auditors and automatically map queries and results into the data structures used by ERP products. This is not as much a technical problem as a problem of usability and expressiveness of the tools.

Outsourcing: the outsourcing of many of the functions performed by a company to external service providers, possible in other countries, makes it harder to audit the part of a business process that is outsourced. This is more of a technical problem than a legal problem, since most outsourcing contracts have provisions for auditing of service providers, and the requirements of compliance frameworks such as Sarbanes-Oxley explicitly extend to service providers.

Lowering the cost of finding fraud: Many companies do not even try to detect fraud because of the upfront cost of buying and configuring software, or developing your own, and staffing the position of internal auditor. The widespread belief (confirmed by many fraud surveys) that fraud is something that only happens to others reinforces this position of denial. Our objective is therefore to lower the TCO (Total Cost of Ownership) of fraud detection tools and improve their detection capabilities such that they pay for themselves, rather than being seen as a cost that has to be bore because of regulatory requirements. This is not an inaccessible goal when one knows that, according to the American Association of Certified Fraud Examiners, companies loose on average 5% of their revenues to fraud.

5 Summary

With the introduction of IT to conduct business we accepted the loss of a human control step. For example, orders would leave the company electronically and fully automated, with no human intervention. For this reason, the introduction of new IT systems was accompanied by the development of the authorization concept. That is why it is of paramount importance that due care is exercised in analysing this system for risks and then minimizing them. This is, of course, also supported by software nowadays. But since, in reality, there is no such thing as total security; auditors are commissioned to examine all transactions for misconduct. Since the data exists in digital form already, it makes sense to use computer-based processes to analyse it. Such processes allow the auditor to carry out extensive checks within an acceptable timeframe and with reasonable effort.

The research community has come up with a number of different approaches over the years to the problem of fraud detection. However, the state of the art of fraud detection by professional auditors is very different. Most of the tools used are simple tools for data import and analysis that leave the detection and classification of potential fraud cases entirely to the human being. In a recent survey, nearly half the auditors who responded reported using generic tools such as Microsoft Excel or Microsoft Access to do their work. This is clearly a sign that the current offering of Computer-Assisted Auditing Tools (CAATS) is not sufficient, and that existing research results have failed to transfer into usable tools for auditors.

We distinguish between generic, specialized and custom-built fraud detection tools. Studying the offers in detail, we realized that the seemingly primitive state of the field can be explained by the fact that most of the advanced tools developed are either custom-built, and therefore very rarely publicized, or are used in very specialized solutions that, for commercial reasons, like to remain discrete on how exactly they achieve fraud detection. Another thing to keep in mind is that the target audience for fraud detection tools, *i.e.*, fraud auditors, cannot be described as tech-savvy. As a result, it is hard to find people with the right combination of computer science expertise and auditing expertise to match the various approaches to the actual problems faced by auditors. We argue that the detection of insider fraud is a similar, yet different problem from that of detecting intrusion at host or network level. A main reason is that attacks against networks are more automated and frequent than attempts at perpetrating fraud though enterprise information systems, which we believe show more variability. The fact remains, however, that we have very little data available to study patterns of fraud in enterprise systems. We present a number of challenges faced today by the designers of fraud detection solutions, which we hope will help steer the community in the right direction. The first challenge is to bridge the gap between existing fraud detection tools and the business level. We need tools that any auditor can use. Another challenge that appeared in the last decade is the increasing outsourcing of non-core functions of a company to external entities. Even if the legal frameworks in place extend auditing requirements to these outsourcing companies, lots of technical and sometimes legal barriers exist to cross-company fraud detection. Finally, and this is more of a business issue than strictly a research one, the enterprise applications industry needs to lower the Total Cost of Ownership of detecting fraud in enterprises. We need to come to the point when installing and using anti-fraud tools pays for itself through the money recovered.

Acknowledgements

The research leading to these results has received funding from the German Federal Ministry of Economy and Technology under promotional reference 01MQ07012 (project THESEUS/TEXO). The authors take the responsibility for their contribution.

References

1. ACFE: Report to the nation. URL http://www.acfe.com/resources/publications.asp?copy=rttn. Accessed on 09/21/2007.
2. ACL. URL http://www.acl.com. Accessed on 09/20/2007.
3. Bay, S., Kumaraswamy, K., Anderle, M.G., Kumar, R., Steier, D.M.: Large scale detection of irregularities in accounting data. In: ICDM, pp. 75–86. IEEE Computer Society (2006)
4. Caseware: IDEA. URL http://www.caseware-idea.com. Accessed on 09/20/2007.
5. Coderre, D.G.: Fraud Detection: A Revealing Look at Fraud, 2 edn. Ekaros Analytical (2004)
6. Coderre, D.G.: CAATTs and Other BEASTs for Auditors, 3 edn. Ekaros Analytical (2005)
7. Coderre, D.G.: Fraud Analysis Techniques Using ACL. Wiley (2009)
8. Economist: Secrets of the digital detectives. URL http://www.economist.com/displayStory.cfm?story_id=7904281. Accessed on 09/21/2007.
9. Insurance Services Offices. URL http://www.iso.com. Accessed on 09/20/2007.
10. Lanza, R.B.: Payables Test Set for ACL. Ekaros Analytical (2003)
11. NetMap Analytics. URL http://www.netmap.com.au. Accessed on 09/20/2007.
12. SAP: GRC Access Control. URL http://www.sap.com/solutions/grc/accessandauthorization/index.epx. Accessed on 09/21/2007.

Combining Traditional Cyber Security Audit Data with Psychosocial Data: Towards Predictive Modeling for Insider Threat Mitigation

Frank L. Greitzer and Deborah A. Frincke

Abstract The purpose of this chapter is to motivate the combination of traditional cyber security audit data with psychosocial data, to support a move from an insider threat detection stance to one that enables prediction of potential insider presence. Two distinctive aspects of the approach are the objective of predicting or anticipating potential risks and the use of organizational data in addition to cyber data to support the analysis. The chapter describes the challenges of this endeavor and reports on progress in defining a usable set of predictive indicators, developing a framework for integrating the analysis of organizational and cyber security data to yield predictions about possible insider exploits, and developing the knowledge base and reasoning capability of the system. We also outline the types of errors that one expects in a predictive system versus a detection system and discuss how those errors can affect the usefulness of the results.

1 Introduction

Imagine this scenario:

> *John, a long-time system administrator at a company has become a problem employee. He's never really caused trouble before, but the manager has received multiple complaints from*

Frank L. Greitzer
Pacific Northwest National Laboratory, Richland, WA 99352, U.S.A., e-mail: frank.greitzer@pnl.gov

Deborah A. Frincke
Pacific Northwest National Laboratory, Richland, WA 99352, U.S.A., e-mail: deborah.frincke@pnl.gov

This work has been supported by the Information and Infrastructure Integrity Initiative, an internal (Laboratory Directed Research and Development) investment of the Pacific Northwest National Laboratory in Richland, WA. The Pacific Northwest National Laboratory is operated by Battelle for the U.S. Department of Energy under contract DE-AC05-76RL01830.

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_5, © Springer Science+Business Media, LLC 2010

coworkers about him showing up late every day. The manager therefore decides to give John a written warning about his attendance last week. After getting the warning, John talks to his manager and loses his cool–storming out of the office–but later calls to apologize. John's manager decides to chalk it up to his recent divorce. The manager also knows that John is probably harboring some resentment after being passed over for a promotion last month; the manager had to have a talk with him after John complained to Human Resources that the manager had been unfair to him. During this time, the Security Office at the company, which checks the local arrest records every day, learns that John was picked up for driving under the influence a few weeks ago.

Fast forward a few months and John is now in dire financial straits because of lawyer's fees both for the DUI and divorce. When a rival company contacted him last year about a lucrative deal if he would obtain guarded company secrets for them, he declined citing loyalty to the company. Now John decides that he has no choice but to entertain the thought—plus, he thinks, "They owe me for not giving me that raise and promotion."

One can observe that John's behaviors suggest a higher risk of a security threat—a disgruntled employee, stressed and angry after going through a major life change and not getting what he thinks he deserves at work. He's got family issues, probably has financial issues, might be struggling with alcohol, and is facing criminal prosecution and embarrassment. We also know he's a trusted system administrator who has the ability to access to private company information.

Mary has been a dependable employee at her company for several years but in the last year she has had difficulties finding projects to fully occupy her time. This stressful situation manifested in her behavior through some subtle and not-so-subtle arguments and complaints from other staff and management; she was eventually reprimanded on her last performance review for erratic job performance. There have even been suggestions about possible layoffs in the coming months.

Stressed about her job security, she decides to carefully engineer a situation that will make her look more indispensible or at least allow her some satisfaction if she is fired: She will sabotage a critical project just before an important demo: if she is fired before the demo it will likely fail; if she is not fired, she can "catch" and "fix" the problem just in time to be a project hero. Mary knows that the project demo uses some legacy software components that required some security patches to be skipped due to compatibility concerns. One of these patches addressed the ability for user processes to insert controlling processes into the system through a simple text editor. Mary realizes that she can exploit this vulnerability by inserting a script that will block critical resources needed for the demo. To disguise her efforts, Mary uses a key logger to acquire the login information of an associate, Bill, who is working on the same project. Using Bill's computer, Mary logs in to the demo computer remotely and installs the script, which triggers several resource-blocking tasks that effectively lock down the demo computer unless a password is input within a certain period of time.

Insider threats represent an especially insidious threat to organizations. As trusted employees, they are given access to information that could compromise the organization if it falls into the wrong hands. Despite much research into the psychology and motivation of insiders, the fact remains that it is extremely difficult to predict insider attacks [18]. This presents organizations with a dilemma. On the one hand, most cases of insider espionage could have been prevented by timely and effective action to address the anger, pain, anxiety, or psychological impairment of perpetrators, who exhibited signs of vulnerability or risk well in advance of the crime [31].

On the other hand, mistaken prediction of potential insider crime may have severe negative consequences for the individuals under scrutiny as well as the organization. Nevertheless, we believe that the potential payoff in avoiding significant asset and information loss serves to justify further research and development of predictive mechanisms involving psychological indicators that appear to have potential in improving prediction. A similar approach to prediction was suggested in 2002 by Schultz [30], involving identification of attack-related behaviors and symptoms—"indicators" that include deliberate markers, meaningful errors, preparatory behaviors, correlated usage patterns, verbal behavior, and personality traits—from which "clues can be pieced together to predict and detect an attack." (p. 526).

Therefore, we advocate research focused on combining traditionally monitored information security data (*e.g.*, workstation/Internet activity) with other kinds of organizational and social data to infer the motivations of individuals and predict the actions that they are undertaking, which may allow early identification of high-risk individuals. This approach is speculative, and it tends to generate privacy debates about whether methods that include monitoring of employee "psychosocial" data to predict or detect insider threats are intrusive and counterproductive: while proactive mitigation may be an advantage to the organization and to the employees who might be helped by timely intervention, it is possible that the practice might be viewed as excessive and invasive by employees. This could exacerbate an already precarious situation and lead to more—or more severe—malicious insider threat events than using less invasive methods alone (which typically do not consider personnel data or at most consider personnel data through traditional management routes). Increased intrusiveness or severity of security measures may contribute to employee job dissatisfaction; management intervention on suspected employee disgruntlement issues may actually increase an employee's frustration level [31]. At the opposite extreme, it is possible that inadequate attention and action can also feed and increase malicious insider activity. One manifestation of this idea has been described as the "trust trap" in which the organization's trust in individuals increases over time, yielding a false sense of security because the trust leads to decreased vigilance toward the threat. This produces fewer discoveries of harmful actions, which in turn increases the organization's level of trust [2, 17].

The purpose of this chapter is to describe a research program that is focused on addressing the research challenges in devising a predictive approach to determining whether an insider threat is present, or whether there is potential for an employee to become an insider threat. The key to prediction as opposed to detection is to incorporate traditional cyber security audit data (normally used to determine policy violations or outlier behavior) with demographic/organizational data about the employee. Clearly this is an area of great sensitivity. By its very nature, prediction is not detection and one would expect that the outcome of any predictive analysis would have a number of gray areas. These would include false positives in the usual sense (someone is falsely identified as a potential insider threat who is not one), false positives in the predictive sense (someone is identified as a potential insider threat who would never become one), false negatives in the traditional sense (an active insider threat is not identified), and false negatives in the predictive sense (a potential in-

sider threat is not identified). Thus, in addition to discussing the model and the need for incorporating such data, we will touch on the ethical and practical dilemmas that come about when seeking to use prediction, and when seeking to incorporate psychosocial data. As part of our discussion, we will report on our progress and results in defining a usable set of predictive indicators, developing a framework for integrating the analysis of organizational and cyber data to yield predictions about possible insider exploits, and developing the knowledge base and reasoning capability of such a system. A more comprehensive discussion of background, approaches to insider threat detection, and organizational challenges is provided in [13, 15].

2 Background

The Threat

Espionage and sabotage involving computer networks are among the most pressing cyber security challenges that threaten government and private sector information infrastructures. Surveys, such as the 2004 *e-Crime Watch Survey* [34], reveal that current or former employees and contractors are the second greatest cyber security threat, exceeded only by hackers, and that the number of insider security incidents has increased geometrically in recent years. The insider threat is manifested when human behavior departs from compliance with established policies, regardless of whether it results from malice or a disregard for security policies. The types of crimes and abuse associated with insider threats are significant; the most serious include espionage, sabotage, terrorism, embezzlement, extortion, bribery, and corruption. Malicious activities include an even broader range of exploits, such as copyright violations, negligent use of classified data, fraud, unauthorized access to sensitive information, and illicit communications with unauthorized recipients.

The "insider" is an individual currently or at one time authorized to access an organization's information system, data, or network; such authorization implies a degree of trust in the individual. The *insider threat* refers to harmful acts that trusted insiders might carry out; for example, something that causes harm to the organization, or an unauthorized act that benefits the individual. A 1997 U.S Department of Defense Inspector General report [38] found that 87 percent of identified intruders into Department of Defense information systems were either employees or others internal to the organization. More generally, recent studies of cybercrime (such as the 2004-2007 *e-Crime Watch Surveys* [34, 35, 36, 37]; see also [17]) in both government and commercial sectors reveal that while the proportion of insider events is declining (31% in 2004 and 27% in 2006), the financial impact and operating losses due to insider intrusions are increasing, and of those companies experiencing security events, the majority (55%) report at least one insider event (up from 39% in 2005).

The Challenge

Currently, no single threat assessment technique gives a complete picture of the insider threat problem. Interviews conducted by our team with several cyber security personnel at our laboratory revealed that the process is generally as shown in Figure 1, but that there is no specific, systematic, and consistent analysis process or use of specific tools. Individual analysts use tools with which they are most familiar, focusing on different types of data.

Current practice in insider threat detection tends to be reactive as it focuses on detecting malicious acts after they occur with the aim of identifying and disciplining the perpetrator. Typical approaches incorporate forensic measures including external threat/defense-oriented appliances such as Intrusion Detection or Prevention Systems (IDS/IPS). There are no tools or visualizations that help the analyst integrate distributed sensor data, or that address information overload by screening data to focus attention on situations that represent the highest potential risk. Tools are largely focused on forensics (after the exploit has occurred) and provide no insight into or anticipatory views of potential insider threat indicators. Thus, the cyber security analysis process could benefit from automated assistance with data synthesis and fusion to improve situation awareness, as well as methods for collecting and assessing possible behavioral/psychosocial indicators.

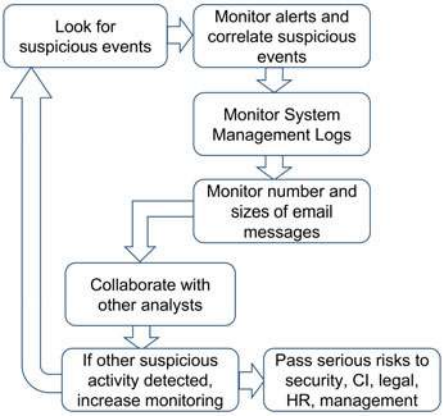

Fig. 1 Cyber-analytic process for insider threat.

The *Insider Threat to Information Systems* state-of-the-art report from the Information Assurance Technology Analysis Center [11] provides a comprehensive review of research being conducted to develop insider threat detection or mitigation models. We review approaches to insider threat modeling in the following paragraphs, but by no means is the following a comprehensive review of all approaches. Wood [39] was the first to devise a comprehensive taxonomy for describing insider threats. The taxonomy is a simple list of attributes: "Access, Knowledge, Privileges, Skills, Risks, Tactics (attack behaviors), Motivation, and Process." Each attribute

is characterized to simulate malicious insider behavior. Magklaras and Furnell [21] devised a broad taxonomy of *misuse*. An important difference for this taxonomy is that it considers *accidental* misuse. A misuse is either characterized as accidental or intentional. We often label an intentional misuse as malicious. The authors observe that accidental misuse can have consequences as large as intentional misuse as seen in several recent examples of accidental misuse that have resulted in large monetary losses. Maybury *et al.* [23] noted that insiders left a trail of cyber activity. From the examination of multiple case studies, they devised a taxonomy of cyber events and their associated observables for detecting malicious insider behavior. Cyber-observable data includes network and system activities, information reconnaissance, accesses to assets, manipulation of assets, and data leakage.

Butts *et al.* [5] apply functional decomposition to devise their insider threat model. Their model appears as hierarchal trees with the root representing the enterprise. The next level of nodes is actions. An insider by way of four *actions*—alteration, distribution, snooping, and elevation—can cause unauthorized changes to the state of the enterprise. Threats are then decomposed into actions with leaf nodes indicating the tools that the insider will use to perform the action. Chinchani *et al.* [8] propose another graph-based model called *key challenge graphs*, a concept similar to attack graphs.[1] The authors state that constructing attack graphs is unwieldy for modeling insider threat because of the lack of automation in threat analysis. The vertices of the key challenge graph represent an entity for which information can be obtained. If the insider visits a vertex, he obtains *keys*, pieces of information that provide for unauthorized access to other entities. A directed edge between vertices represents an access or communication channel that facilitates insider actions. The edges are labeled with a *key challenge*, a relative measure of the quality of access control. A key challenge is a triple consisting of a key, the cost of accessing the entity with the key, and the cost without the key. As such, multiple edges will exist between pairs of vertices to indicate the total cost of visiting a vertex.

In *system dynamics* [6], a method of analyzing the behavior of complex systems over time, simulations are used to model the complexities of the insider threat problem. System dynamics is used similarly in [2]. The variables used in the model were structured on observable events and behaviors that permit early detection of insider aberrant behavior. Interestingly, the model includes variables for psychological stresses—such as organizational sanctions and punitive actions—that may trigger insider actions.

The Detection of Threat Behavior (DTB) project [9] uses a data mining application and a Bayesian network to detect aberrant document access patterns. Aleman-Meza, Burns, and Eavenson [1] examined the use of semantic associations within an ontology for capturing the scope of a cyber investigation and for determining document relevance to an investigation. The authors state that the associations expressed in the ontology are better at capturing heterogeneous domain semantics than either clustering or automatic document classification.

[1] Attack graphs represent the ways in which an adversary can exploit vulnerabilities to break into a system.

Magklaras and Furnell have developed their Insider Threat Prediction Model (ITPM) [21, 22]. The model computes a single-value score from a set of user threat factors such as user sophistication, role, and access to assets. The score is then used to place the user in one of four threat categories: possible internal threat, potential accidental threat, suspicious, or harmless.

More psychological and organizational/management research (*e.g.*, [2, 12, 17, 19, 25, 27, 29, 31]) suggests that a proactive approach must recognize possible precursors to malicious insider threat behavior that are manifested in employee stress, disgruntlement, and other signs that we refer to as psychosocial factors. No current system integrates cyber and psychosocial data into a predictive framework; our research seeks to develop such a framework.

3 Issues of Security and Privacy

A core concern is the tension between the needs of an organization to safeguard its assets and the privacy rights and expectations of individuals who use organizational resources. Greitzer, Frincke, and Zabriskie [13] discuss this complex matter in detail; in this chapter we address the issue from the perspective of its implications for models of insider threats that incorporate psychosocial data.

The notion of privacy is generally characterized in terms of the right of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated. There is a fine line between what the organization needs to know and what is firmly in the realm of the employee's expectation of privacy. Few employees actually engage in activities that would constitute insider threat; the rest of the population consists of honest, hard-working staff, some of whom would be highly offended to learn they were being monitored.

Issues of security and privacy are founded mostly upon law and ethics. With regard to legal foundations, there is no single source of privacy law in the United States, but rather an incomplete patchwork that includes the U.S. Constitution, the Fourth Amendment to the U.S. Constitution (protecting against unlawful search and seizure), State constitutions, and federal and state statutes (the reader is advised that many U.S. states have more restrictive laws and requirements too numerous to discuss here [2]). Workplace privacy has been a regularly contested area of law and policy. The preponderance of opinion in court decisions has shifted over time to the position that ownership of servers by business organizations supersedes claims of privacy by employees who use computer systems as part of their jobs. This means that employers may routinely monitor employee e-mails and workstation activities (*e.g.*, web-surfing). Although there are laws that prevent an employer from sharing intimate employee information with individuals outside the company, there are few restrictions on an employer's right to share it with people on the inside [20]. Notice

[2] For a compilation of federal and state privacy laws, see R. E. Smith (2002) Compilation of State and Federal Privacy Laws. [with 2009 Supplement ISBN: 9780930072179], Published by Privacy Journal, P.O. Box 28577, Providence, RI 02908.

by the employer of such monitoring practices generally defeats an employee's "reasonable expectation of privacy." [3] Indeed, mere use of an employer's email system may mean an employee has no reasonable expectation of privacy in email; and even if the employee owns the computer and brings it into the workplace, this does not necessarily establish a reasonable expectation of privacy.[4] Thus, employers monitor computer use of employees by tracking the use of Internet as well as internal information systems and, to a varying degree, by screening email metadata and contents. It is widely acknowledged [5] that employers have the right to monitor employee computer activities and to internally share personal employee data, although federal agencies can use this information only for job actions if "performed for foreign counterintelligence purposes or to produce background checks for security clearance of Federal personnel or Federal contractor personnel." [6]

The ethics foundation of the debate about the issue of employee monitoring is ultimately based on the concept of trust. Trust has been described as the belief in, and willingness to depend upon another party [24]. Tabak and Smith [33] describe the implications for workplace monitoring from the perspective of trust initiation and formation between management and employees. They describe the development of trust in terms of a "sensemaking" process that interprets an individual's current work experience (external information consisting of issues, events, objects, and individuals in the environment) in relation to previous work experience (in the form of existing knowledge structures or schemas). The relationship between an individual's current and past work experience influences the individual's disposition to trust. In this view, the initiation of trust and subsequent trust formation affects managerial implementation of electronic monitoring policies, and these policies have implications for workplace privacy rights. Similarly, these same factors influence trust formation by employees, and management practices such as workplace monitoring are perceived by employees in ways that influence employee trust in and commitment to the organization.

To some employers, the cost and damage of one instance of sabotage or espionage might in certain cases warrant monitoring behavioral and demographic employee data available to anticipate and prevent incidents. From this perspective an employer might argue that monitoring promotes productivity and affords better control over employees; and is justified because employers pay for employee time and own resources such as computer equipment and network connections. And, perhaps an even more gray area, that when on the job, an employee's observable and relevant personal behavior should be taken into consideration as a risk factor. This is not a universally popular perspective, nor is it universally accepted as a way to achieve the employer's goals. Critics note that electronic workplace monitoring can increase

[3] Biby v. Board of Regents, 419 F.3d 845 (8th Cir. 2005); and TBG Ins. Servs. Corp. v. Superior Court, 96 Cal. App. 4th 443, 452 (Cal. Ct. App. 2002).

[4] United States v. Barrows, 481 F.3d 1246 (10th Cir. 2007); and United States v. King, 509 F.3d 1338 (11th Cir. 2007).

[5] See for example `http://www.salon.com/tech/feature/1999/12/08/email_monitoring`.

[6] See `http://www.usdoj.gov/oip/privstat.htm`.

employee stress, reduce commitment, and lower productivity [4]. They also point out ethical implications relating to employee perceptions of their own privacy rights and the possible impacts on their sense of well-being and quality of work life [29]. Privacy rights advocates are concerned that potentially harmful information about individuals or their loved ones may be subject to unwanted intrusions. Extensive monitoring that is perceived as invasive may contribute to an employee's job dissatisfaction. Management intervention on suspected employee disgruntlement issues may actually increase an employee's frustration level [33]. Moreover, false accusations that can arise from predictive approaches may affect the career of the accused. Complicating the situation further, inadequate attention and action by an employer can *increase* insider activity. Such influences on trust have been described as the "trust trap", *e.g.*, [2].

The privacy and ethics debate is clearly a contentious issue that deserves more discussion by the research community and by stakeholders from government, industry, and the public [13]. Employment is founded upon trust that employee/employer will meet certain expectations, ranging from exchanging work for a paycheck to treatment within the organization. The ability of an employee/employer to have a trusted relationship depends on many factors, including privacy, individual rights, rights of the organization, and the organization's power. While an organization may assert—and in many cases society acknowledges—its right to conduct electronic workplace monitoring, there is also the potential for negative backlash (reduced trust). This is especially true if the organization imposes monitoring surreptitiously or without advance notice. If the process is fully disclosed and explained, as well as managed equitably, it is not as likely to be considered unfair by employees and the mutual trust relationship required for a healthy organization is more likely to remain intact. The monitoring then becomes a known and understood element of the conditions of employment.

We suggest that any data monitoring needed to inform a predictive (and behaviorally based) model should be completed openly and with proper privacy safeguards and should be based on actual behavior and events that are identified in a fashion similar to the normal performance assessment process. The sensitive nature of the data collection and analysis that we are considering for insider threat monitoring suggests that additional discussion and vetting of the process are needed before it will be possible to establish an infrastructure for employee monitoring of this type. Performance reviews are generally acceptable when they are perceived to be administered fairly; perhaps also because they do not disrupt the established processes of an office, or because they are often defined as career development activity (rather than performance evaluation per se). If the standard practice of performance evaluation is not considered an invasion of privacy, then would not the similar practice of assessing factors that may affect performance (such as stress, disgruntlement, etc.) also satisfy privacy criteria? From both an employee/career development perspective and considerations of workplace productivity and morale, it could be argued that enabling managers to assess motivational factors should bring about benefits to both the organization and the employees, as long as (a) managers have appropriate

skills to recognize potential problems and (b) they have requisite interpersonal skills and appropriate policies to mitigate problems.[7]

Other concerns that have been voiced about monitoring psychosocial factors of employees focus on possible negative effects on an employee's job security or performance evaluation if monitoring leads to elevation of risk, even though subsequent analysis reveals no threat activity. Clearly if such were the case, the employee would have a basis for a complaint about improper practices. There should be safeguards in place against such unfair outcomes; some are already legislated: *e.g.*, employee financial and health records are required to be stored in separate data repositories from personnel files.

4 Predictive Modeling Approach

Our predictive approach has been developed under a three-year internally funded research and development program at PNNL [15]. The model, embedded in a prototype system called PsyberSleuth, addresses the threat indicators preceding the exploit. Some of these indicators may be directly observable, while others are inferred or derived based on observed data. We recognize the imperative to address both human and cyber elements that compose the insider threat. Defining possible precursors in terms of observable cyber and social/organizational (psychosocial) indicators and developing an analytic methodology that integrates psychosocial and cyber indicators is one of the major challenges in developing a predictive methodology.

Conceptual Model for Predictive Classification

At the highest level, the model comprises a knowledge base of indicators and heuristic models of insider behavior. Indicators are essentially the semantics of insider behavior and characteristics—interpretations of intentions and actions based on observations. This knowledge base informs all of the components of the insider threat model, and is in turn updated or modified by outputs from components that perform functions such as data collection, data fusion, and analysis. The process can be thought of as a multi-layered analysis/inference process that progresses from *Data* to *Observations* to *Indicators* to *Behaviors*, as depicted in Figure 2. A description of these processing activities is as follows:

- *Data* are processed to infer *observations*.

  Examples: Data representing activity of an employee's network account such as Web traffic/outgoing or incoming data through firewall per Internet Protocol account mapped to domain login. An algorithm may calculate the amount of such Web traffic over time and compute trends in amount downloaded or uploaded, or ratio of upload to download

[7] The conundrum is that good management engenders maximum trust, while management that engenders distrust just is not good management.

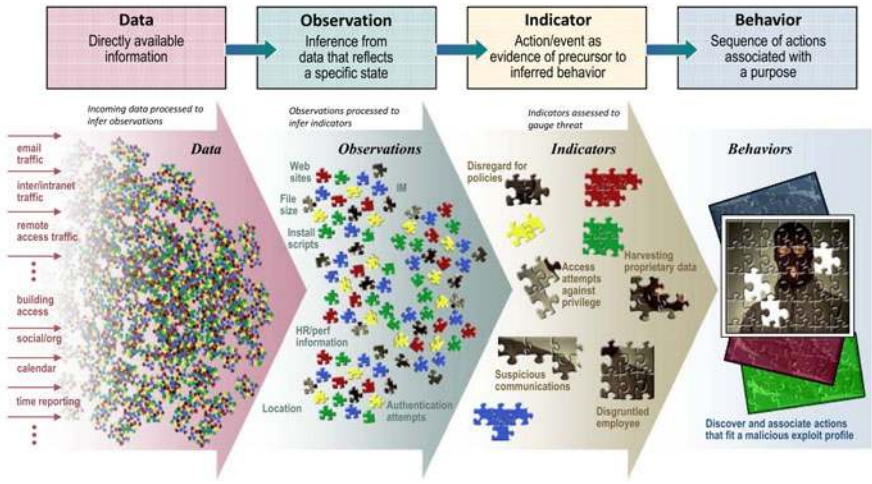

Fig. 2 Model-Based Predictive Classification Concept: Incoming data processed to infer observations; observations processed to infer indicators; indicators assessed to gauge threat.

and track changes in such variables over time. Resultant figures may be considered observations such as "amount of Internet Web surfing" or "amount of downloads." Other examples: observations such as time at work or hours worked may be derived from data such as timecard records, Virtual Private Network (VPN) login, etc. The observation "screen saver disabled" may be derived from data such as the presence/value of a screen saver registry key that indicates if/when the screen saver is enabled on a machine.

- *Observations* are processed to infer *indicators*.

  Examples: From observations relating to Web traffic (see above), indicators may be derived that represent "possible suspicious" activities such as "increased downloads above normal" or "unusual/late hours worked." Other examples of indicators based on related observations are "unusual amount of internet web-surfing," "excessive attempts to access privileged data base," "presence of automated scripts" and "use of personal email account." On the psychosocial side, we may identify an indicator such as "anger management issues" based on data and observations such as entries in a management or human resources (HR) database relating to arguments with supervisors; or "disgruntled employee" that represents a staff member who exhibits various manifestations/indicators of anger in the workplace.

- *Indicators* are examined to infer *behaviors*. Behaviors are sequences of activities for achieving some specific purpose that can be characterized as malicious or benign. We are most interested in malicious behaviors that are consistent with established patterns or profiles exhibited in insider exploits.

  Examples: Malicious behaviors are combinations/sequences of indicators or behaviors, such as manifestations of abuse or attempting to circumvent policy by accessing data without privilege. A malicious exploit that involves theft of intellectual property would be represented by a collection of indicators or behaviors defined within the knowledge base, with some temporal constraints when the order of such sequences is important.

Another conceptual view of the modeling framework is illustrated in Figure 3, which shows components of the predictive classification model that perform data collection, data fusion, analysis, and decision-making functions that must recognize predictors of malicious exploits, assesses risk levels, and formulate responses/actions to mitigate the risk.

Monitoring/Data Collection Components

A fundamental assumption about the data collection component is to recognize that not all possible data can be collected all or even some of the time. More data is not necessarily desirable. Thus, heuristics and other considerations might modulate the nature and quantity of data that are collected and retained in data repositories. A second fundamental notion is that not all of the data that are collected are necessarily weighted equally over time. The determination as to whether or not to collect/analyze more data (or to conduct more detailed or sophisticated analysis on existing data) is based upon the level of suspicion that is computed for the current phase of analysis. We acknowledge that some observations (or derived indicators) may yield an immediate response because they are immediately recognized as malicious acts. On the other hand, we expect that the most dangerous insiders operate in a more subtle manner and attempts to hide behaviors within the "noise" so that it is difficult to detect. These "low and slow" or gradual changes in cyber behavior are

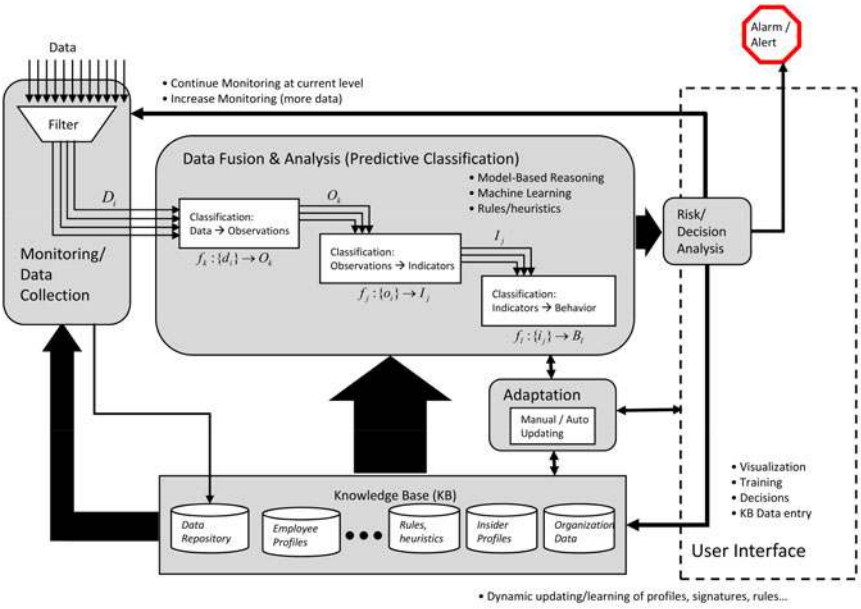

Fig. 3 Predictive/Classification Framework.

generally more sophisticated. The techniques used enable the bad actor to operate "below the radar" and thwart detection systems that are based on simple detection thresholds.

Knowledge Base Components

The knowledge base describes the insider threat problem domain. Populating the knowledge base with specific definitions of exploits is key to achieving the level of performance that is needed to implement a deployable system.

It is important to distinguish between this predictive classification model and current approaches that rely exclusively on pattern recognition and anomaly detection. Our conceptual model employs a hybrid approach that is based on pattern recognition processes but not merely dependent on identifying discrepancies from "normal" behavior as the primary means of threat analysis. Rather, the knowledge base is populated with scenarios or behavioral templates that reflect possible malicious exploits. While deviation from norms is considered as part of the analysis, so is conformance with prototypical exploits and behaviors that have been identified, through extensive research, with malicious intentions and actions. The challenge is to conduct model-based *reasoning* on the recognized patterns at a higher semantic level of concepts/constructs, rather than applying fixed recognition processes to fixed signatures.[8] This is accomplished through sophisticated reasoning components.

Reasoning Component

The predictive/classification components analyze a vast amount of noisy data that are in a constant state of change and produce outputs corresponding to classification of possible precursors or indicators of malicious intent or behaviors. This output must be interpreted in terms of risk or level of belief that a malicious exploit is being planned or under way. The executive function, or *Reasoner*, has this responsibility. The Reasoner operates over the insider threat problem domain described by the knowledge base. Incoming data are directly available either through cyber monitoring or from databases that contain organizational/staff development information.[9]

---

[8] The problem is essentially identical to the early discussion of human cognitive/perceptual systems that argue against a pure template-recognition model that would have to store an essentially infinite number of variations of a concept (such as for "chair") rather than its attributes and functional features or behaviors (one can recognize a chair even if that particular instance has never been seen before).

[9] The reader may ask, "How and where are such social/organizational data collected and maintained?" We recognize that infrastructures for collecting and maintaining such data are not likely to be present in most organizations; however, conversations with HR personnel suggest that such information, while not formally tracked in a single database, are generally known to managers and

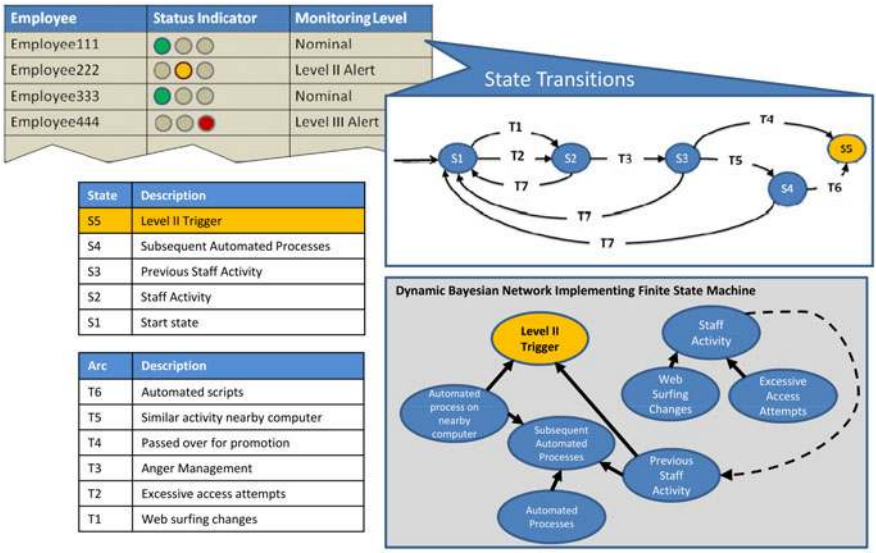

Fig. 4 Illustration of the Reasoner output and its representation as a finite state machine for the portion of the model relevant to the example scenario, showing states (S), transitions (T) among states, and implementation of the finite state machine as a dynamic Bayesian network.

The Reasoner processes data to infer cognitively meaningful states, or observations. The degrees to which the observations are supported are recorded as *virtual evidence* [28] on nodes in a Dynamic Bayesian Network (DBN). From these, the DBN calculates belief levels assigned to *indicators*. In many scenarios, the order of events matters, as does the elapsed time between events. These temporal properties are captured by finite state machines modeled by the DBN. The Reasoner next assesses current indicators in combination with previously inferred indicators and behaviors to determine the likelihood of behaviors that represent threats.

Figure 4 illustrates the process. At top left is a conceptual representation of the momentary status of the employee list. A three-level status hierarchy is assumed in which baseline or nominal status applies initially to all employees, after which monitoring and analysis as described above may elevate the status of an employee to a higher risk level. In the figure, *Employee222* represents a case that is similar to the example scenario at the beginning of this chapter.

The finite state machine is implemented within the DBN by having states and transitions represented by nodes. The transition from state S_i to state S_j via transition T_k is modeled by calculating the current belief for S_j from the previous belief for state S_i and the current belief in transition T_k. Figure 4 illustrates how the previous belief level for *Staff Activity* is used to influence the current belief in *Level II* activity through the placeholder node *Previous Staff Activity*—the dashed line indicates the

human resources personnel. Therefore, we have adopted an assumption that an organization that adopts the recommended approach will also provide the necessary infrastructure for such data.

temporal link. The Reasoner has the ability to age, or partially discount, evidence provided by previous time steps.

The processing/analysis framework is based, in part, on previous work used to analyze scenarios characterized by types of motion[10] [10]. Data are directly represented in the monitored cyber data or available in human resources/security databases. Observations, which reflect cognitively meaningful states that are inferred from data, may be represented as Description Logic classes within the knowledge base. The Reasoner processes observations of class membership to infer indicators. Indicators are described to the Reasoner using Description Logic formulae [26]. Behaviors—modeled as finite state machines—are recognized with associated threat and confidence levels by the Reasoner, which assesses current indicators in combination with previously inferred indicators and behaviors to determine the likelihood of behaviors that represent a threat.

Causal relationships between data, observations, indicators and behavior are tenuous and uncertain. While it is possible to capture expert knowledge about rules relating data to indicators, the set of rules will be incomplete because the knowledge acquisition task is too daunting to capture all the relationships, and the threat environment is constantly changing. This challenge is met in several ways. First, the knowledge elicitation process is facilitated by encoding expert knowledge using fuzzy sets [40]. Thus, experts can specify, for example, that an individual who is "very unhappy" with his or her job situation represents a potential insider threat. In addition, experts can specify how data should be interpreted as evidence for particular indicators, using probability values to indicate the strength of the evidence. Perhaps only 5% invalid login attempts represent attempts to access data, while a log record indicating a breach represents a problem in almost all cases. Finally, by using description logics as our knowledge representation, we allow general statements to be made about classes of behaviors or indicators while supporting more specific descriptions of well-defined terms.

Data Sources

Examples of computer workstation activities at the *data* level (cyber data) include registry entries, a variety of logs (event logs, firewall logs, network IDS logs, Domain Name Server (DNS) logs, database server logs, query logs, etc.), file permissions, email headers, detection of malware, and also perhaps even more intrusive data such as keystroke records and email content.

Examples of social/organizational data include annual performance review input; performance awards/recognition; clearance/background check information if available; assessments of managers and/or peers about the employee's social/organizational behavior such as disgruntlement, anger, and stress; physical security access; attendance records/time reporting records; etc. Because psychosocial data used in

[10] While the previous work extracted data from video streams, its primary focus was a semantic understanding of the content, which was analyzed in terms of domain knowledge related to the steps used to implant an improvised explosive device.

the analysis are of practical and theoretical interest in their own right, this aspect of the development effort is described separately in the next section.

It is clear that the availability and update frequency of cyber data will be much higher than that of social/organizational data. In some cases the latter data may have to be entered manually into a database in order to be accessible by an insider threat system, although for our purposes it is assumed that a future deployment would have such data available. One aspect of a model that integrates cyber and psychosocial data is not only the difference in quantity and update frequency, but also the fact that psychosocial data may become known asynchronously: unlike cyber data, psychosocial data are not acquired in real time. It is conceivable, for example, that a critical psychosocial event may become known weeks or months after it actually occurred. This has a profound implication for the model architecture: the reasoning system must accommodate acquisition of new data that can change the previous outputs of the Reasoner. In essence, this implies that the entire set of data must be analyzed at each analysis cycle.

Psychosocial Factors and Indicators

Elsewhere [13] we have discussed a variety of demographic, behavioral, or psychosocial data that, in various combinations, may provide warning signs of malicious insider threats. Based on case studies and research already cited here, these signs potentially may be used as risk indicators for insider threat. As a result of a discussion and consideration of legal and ethical boundaries, Greitzer *et al.* [13] developed a preliminary set of indicators that (in our assessment) do not yield privacy or ethical concerns; and we identified a set of indicators (and associated data sources) that appear to be acceptable, as well as a set of indicators/data sources that are not appropriate.

For example, on legal/ethical grounds, we concluded that the following types of data should not be included in an insider threat monitoring framework: arrest records; use of Employee Assistance Program (*e.g.*, for family counseling); use of Employee Complaint Mechanism; life events (such as marriage, divorce, births, or deaths in family); and health events (medical records). On the other hand, we identified five data sources that appear to show promise in assessing relevant psychosocial factors and that also seem to be reasonable candidates to be considered from a legal/privacy ethics perspective. These are: 360 Profiler and other tools that are used in staff performance evaluations; competency tracking; disciplinary tracking; timecard records; proximity card records; and pre-employment background checks. These sources, by themselves, do not constitute the psychosocial factors directly, but they do inform such factors.

We believe that these appropriate information sources, along with direct observations, are reasonable factors to be considered by an organization. In particular, they could be used by managers and HR personnel to identify when there are grave concerns about psychosocial factors that are considered predictors of potential malicious insider activity—and in some cases already are (disciplinary tracking, for

instance). These indicators are shown in Table 1. Use of these indicators assumes an observational/management reporting approach that would rely on personnel data and judgments that are likely to be available from management and HR staff. Also, it assumes some manner of quality control and possibly employee appeal and review, to reduce likelihood of misuse. Each of these indicators may be described using examples or "proxies" that are more readily observed than psychological constructs identified in the research literature (such as antisocial personality disorder or narcissism) that typically would not be available. In our judgment, these indicators reflect the psychological profiles and behaviors that have been observed in case studies to correlate with insider crime— *e.g.*, personal predispositions that relate directly "... to maladaptive reactions to stress, financial and personal needs leading to personal conflicts and rule violations, chronic disgruntlement, strong reactions to organizational sanctions, concealment of rule violations, and a propensity for escalation during work-related conflicts" [2]. It is important to note that use of any one of these factors in a predictive model would be restricted to cases in which the employee's behavior is considered to be of highest concern. Because these are subjective judgments, only the most severe instances would be propagated in the predictive model.

The psychosocial indicators shown in Table 1 were developed by obtaining judgments from available HR experts on the prevalence and severity of different combinations of indicators that reflect different scenario cases. As revealed by this knowledge-engineering process, these psychosocial indicators contribute differentially to the judged level of psychosocial risk—*disgruntlement*, *difficulty accepting feedback*, *anger management issues*, *disengagement*, and *disregard for authority* have higher weights than other indicators, for example. There are several points that should be emphasized:

- the indicators need to be empirically tested or vetted with larger samples of HR experts and managers to assess their validity, at least at a subjective level;
- the judgments based on observations will necessarily always be subjective—there is no expectation that an objective test instrument will emerge from this research;
- nevertheless, with appropriate training we believe that management and HR personnel would better understand the nature of the threat and the likely precursors or threat indicators that may be usefully reported to cyber security officers;
- most importantly, the approach in predictive modeling is to provide "leads" for cyber security officers to pursue in advance of actual crimes, without which they would likely have little or no insight with which to select higher-risk "persons of interest" on which to focus analyses.

For security analysis purposes, only cases in which a manager is "highly concerned" about such factors or combinations of factors would be advanced in the predictive model to raise the level of concern or risk. As the risk level increases, so too would the level of monitoring and analysis on an individual increase.

Indicator	Description
Disgruntlement	Employee observed to be dissatisfied in current position; chronic indications of discontent, such as strong negative feelings about being passed over for a promotion or being underpaid, undervalued; may have a poor fit with current job.
Accepting Feedback	The employee is observed to have a difficult time accepting criticism, tends to take criticism personally or becomes defensive when message is delivered. Employee has been observed being unwilling to acknowledge errors; or admitting to mistakes; may attempt to cover up errors through lying or deceit.
Anger Management Issues	The employee often allows anger to get pent up inside; employee has trouble managing lingering emotional feelings of anger or rage. Holds strong grudges.
Disengagement	The employee keeps to self, is detached, withdrawn and tends not to interact with individuals or groups; avoids meetings.
Disregard for Authority	The employee disregards rules, authority or policies. Employee feels above the rules or that they only apply to others.
Performance	The employee has received a corrective action (below expectation performance review, verbal warning, written reprimand, suspension, termination) based on poor performance.
Stress	The employee appears to be under physical, mental, or emotional strain or tension that he/she has difficulty handling.
Confrontational Behavior	Employee exhibits argumentative or aggressive behavior or is involved in bullying or intimidation.
Personal Issues	Employee has difficulty keeping personal issues separate from work, and these issues interfere with work.
Self-Centeredness	The employee disregards needs or wishes of others, concerned primarily with own interests and welfare.
Lack of Dependability	Employee is unable to keep commitments /promises; unworthy of trust.
Absenteeism	Employee has exhibited chronic unexplained absenteeism.

Table 1 Suggested Psychosocial Indicators

Validation of Predictive Model

How should the effectiveness of an automated insider threat tool be assessed? No standard metrics or methods exist for measuring success in reducing the insider threat—this "capability gap" is one reason why the insider threat problem was listed in the 2005 INFOSEC Hard Problems List [16]. Other challenges are the lack of appropriate data and "ground truth" for predictive assessment, the large degree of overlap between observable behaviors of normal versus malicious activities, and the difficulty of finding population base rates.

The most rigorous form of evaluation of a predictive model is to test the predictions against a set of real cases, but due to the nature of the problem, applicable cases are rare. Further difficulties arise from the fact that data are collected over long time spans, making it difficult for experts to comprehend and reason about large volumes of data. Experts also may vary in their assessments of risk for a given set of indicators, depending on their background and experiences. In addition, while it is

reasonable for experts to validate the findings of the system to perceived matches to insider threats, it is not practical for experts to examine all the observable data for monitored subjects to determine which of them should be flagged (that in itself might threaten employee privacy!). A confounding problem is that experts could find evidence of a threat that is not modeled by the system, causing difficulties in the interpretation of test results. Finally, in integrating psychosocial indicators with cyber indicators, the model requires experts from disciplines typically outside of the experience and comfort zone of cyber security and counterintelligence analysts.

While an empirical test is the ultimate aim, other intermediate evaluation approaches can be used to test aspects of the model. An objective in validating the psychosocial component of the model was to demonstrate agreement between the model and expert judgments. This requires the following steps:

- Obtain expert judgments on what constitutes a valid threat, what constitutes valid indicators for that threat, and how to tie indicators to observables.
- Develop test scenarios with experts' help—scenarios must be specified in detail with appropriate data and observables that will drive the model.
- Obtain expert judgments on the scenarios that will be used to test the model.
- Operate the model on the data or observables associated with a scenario. The model must characterize the extent to which the observables match a scenario. These outputs are then compared to experts' assessments of the same sets of observables.

Verification has been accomplished by soliciting judgments from expert evaluators who examined the same observables used by the model. Developers conducted this verification both as unit tests and as a quality control on completed models. Evaluations used case studies, similar to but more detailed than the example scenario, and additional datasets (some of which were fabricated to test specific aspects of the model, and some generated by simulation software).

Verification/Validation of Psychosocial Model

The Bayesian network model was developed from two HR experts' judgments—verified by examining the model's agreement with their judgments. Figure 5 (a) shows the results of a verification test comparing the output of the psychosocial model with the (combined) judgments of our HR experts (risk judgments were provided on a 0-10 scale and then normalized to a 0-1 scale). The high value of $R^2 = 0.94$ indicates a good fit.

The model was next validated against the judgments of three additional HR experts who were asked to judge the level of risk in 50 case studies involving different combinations of observed indicators. (This was a subset of the cases used to develop the model.) Figure 5 (b) shows a scatterplot comparing the mean expert judgments with the psychosocial model's output. The resulting correlation was substantially less ($R^2 = 0.29$), although still significant. Much of the disagreement was traced to four cases in which five of the lesser indicators were present without any of the

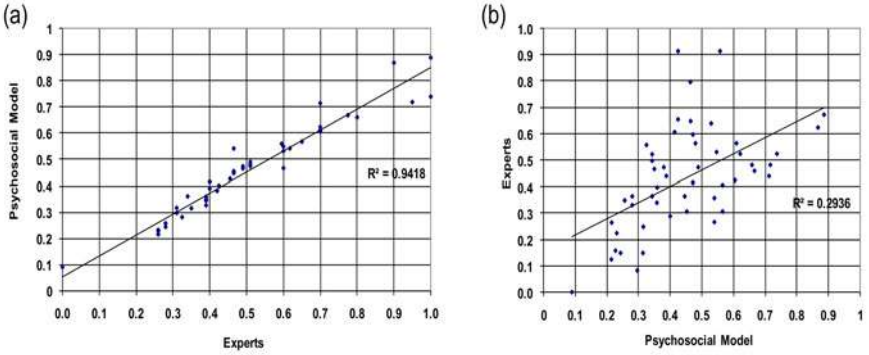

Fig. 5 (a) Verification test showing the fit of the psychosocial model to judgments of two experts used during development of the model ($R^2 = 0.94$); (b) validation test showing the fit of the model to combined judgments of three different experts ($R^2 = 0.29$).

more serious indicators—these were judged differently by our "development" experts compared to the evaluation experts. Further investigation also suggests that the absence of more serious indicators may have been interpreted differently by the two groups of experts. Removing the four cases that did not exhibit any of the most serious indicators, the agreement between pairs of expert evaluators (ranging from $R^2 = 0.33$ to 0.54) was comparable to the agreement between the model's output and the individual evaluators (ranging from $R^2 = 0.21$ to 0.43). Additional work and more careful instructions to expert evaluators about interpreting the indicators will be necessary to calibrate the parameters of the model. At this stage, we may observe that the real-world diversity of expert evaluators can be reflected in the model. It will be valuable to preserve this diversity and allow different analysts to supply different weights or parameters when working with an operational system. This diversity also underlines evaluation challenges noted earlier.

Injection Testing (Verification) of Cyber Component

Because of the complexity of models and variability of evaluators' responses, we employed fault injection [3]. The focus was to determine if the number of detections (defined as an individual whose behavior merits additional monitoring) increased with the number of simulated "bad-guys" (the injection rate).

For the injection study, data were generated for 100 simulated employees over 4 months. Simulation runs were developed for three conditions: 0, 6, and 12 simulated employees were stochastically defined as malicious insiders who would (with elevated probabilities for specific actions) engage in an exfiltration scenario involving an attempt to access and steal privileged information. The simulations were developed by employees who were not involved in implementing the Reasoner. Four simulation runs were conducted at each of the three injection rates—thus the model analyzed a total of 400 simulated employees at each injection rate. Each of the

100-employee simulation sets comprised 2.8-3.3 million records that reflected 39 different kinds of "events" with the present analysis focusing on 10 of these event types for the specific exploit.

For each simulated employee, the model produced output ranging from lowest (0.0) to highest possible (1.0) threat value. We then assessed the association between the observed frequencies of outputs in five threat intervals (0.0-0.2, 0.2-0.4, 0.4-0.6, 0.6-0.8, and 0.8-1.0) at the three injection rates. As would be expected, most of the simulated cases analyzed by the model yielded negligible threat values (roughly two-thirds), with the remainder (406 cases out of 1200) leading to non-zero threat values. Our evaluation focused on these cases to "tease out" the model's outputs for the more challenging data. For these data, a Contingency Coefficient analysis [32] uses a chi-squared significance test with 8 *df*, that is, (5 rows - 1) x (3 columns - 1) = 8, yielding $\chi^2 = 575.3$ $(p << .001)$. The degree of association between the two variables is obtained from the equation, $C = \sqrt{\chi^2/(N+\chi^2)}$, where $N = 406$. The obtained value of $C = 0.76$ indicates a strong association. An employee was deemed "suspicious" if the threat value exceeded a threshold.

Figure 6 plots the cumulative frequencies of detections of staff, above a given threshold (threat value interval), for each of the threat value intervals at each of the three injection rates (data are transformed to frequencies per 100 to make the relationships easier to interpret for the three injection rates). The figure shows that at the injection rate of 12, threat values above 0.8 were found for five staff; 20 staff had threat values above 0.6. If we consider 0.7 as a first approximation of a threshold for "flagging" staff for further analysis, we estimate that eight staff would be "flagged" for the injection rate of 0, six staff would be "flagged" for the injection rate of 6, and 12 staff would be "flagged" for the injection rate of 12. This clearly shows a trend that is consistent with the injection rate. However, the outcome for the zero injection rate condition is problematic, indicating some false alarms. It should be noted that since the flagging represents only a recommendation to increase scrutiny, this is not the same as the traditional false-positive rate associated with detection strategies. Further, the recommended additional scrutiny could reduce the threat value for simulated staff in the zero injection rate condition, thereby reducing false alarms.

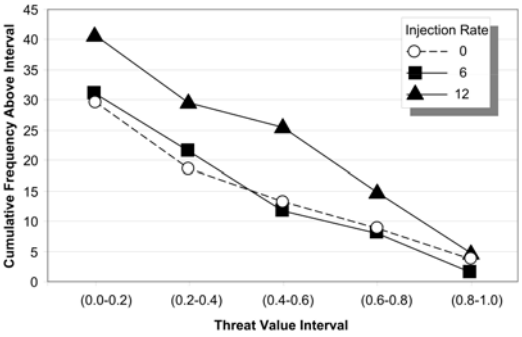

Fig. 6 Cumulative frequencies above a given threat value interval as a function of injection rate.

More challenging evaluations would assess the impact on the operational environment by tracking performance (*e.g.*, comparing system alerts with expert judgment) and morale.

5 Training Needs

In this chapter we have described an approach to insider threat prediction and mitigation that integrates psychosocial and cyber/workstation data. We have discussed privacy and ethical issues that influence the selection and analysis of psychosocial data to protect the privacy of employees while safeguarding the organization's physical and intellectual assets. The success of an approach that integrates psychosocial data will be dependent not only on the technical capability of the model framework but also on the ability of management and HR personnel to identify behavioral precursors and indicators that are used by the predictive modeling framework. It has been noted that typical organizations do not currently employ a reporting/monitoring infrastructure to support such an endeavor. In addition to such infrastructure, it is necessary to educate, train, and raise awareness of employees about the insider threat. Management, HR, and cyber security personnel require training to recognize psychosocial indicators reflected in Table 1, for example, to support detection and early recognition of malicious insiders. All employees should receive training on good security and information technology practices to reduce the risk of inadvertent (non-malicious) actions that provide opportunities for outside exploitation of internal personnel and resources.

Awareness Training for Insider Threat

Greitzer *et al.* [14] discuss the need for training and awareness to combat insider threats. Current efforts to raise awareness about insider exploits, vulnerabilities, and strategies for reducing risks include workshops and computer-based training programs developed by US/CERT [7, 6, 25]. The MERIT training program offered by CERT and based out of Carnegie Mellon University's CyLab, focuses on raising awareness of factors (such as security policies and management practices) that influence employee morale as well as the influence of employee attitudes on management approaches and development of policies. This complicated set of inter-relationships among such factors is captured in system dynamics models, which form a foundation for the MERIT training approach offered by CERT.

When organizations implement countermeasures, unintended consequences may result. The MERIT project describes the severity of the insider threat problem using system dynamics models based on empirical data. Both CERT and MERIT projects found that "intuitive solutions to problems with employees often reduce the problem in the short term but make it much worse in the long term" [2]. Models of the IT sabotage problem were developed in these studies. One example of addressing insider

threat that has unintended consequence is the termination of an employee, which may solve an immediate problem, but subsequently a threat still exists if systems were compromised before termination, which may result in long-term problems for the organization. The hope of studying causal structures of behavioral problems is to understand and communicate the nature of the problem and the benefits of alternative mitigations before the malicious activity takes place.

Industry best practices for preventing and mitigating insider threats are largely policy-centric. In addition, technical best practices aimed at the insider threat differ little from those aimed at external threats, except in that the information defense barriers are inverted. Organizations must take into consideration the legal system and privacy rights when taking both active and passive measures to enforce the protection of proprietary information. Security experts urge corporations to develop an acceptable usage and monitoring policies that balance ethics and best practices in employee monitoring and productivity [11]. Policy and procedural changes may cause issues with morale as well as performance, both of which are already psychological or situational indicators of a potential threat problem. The use of system dynamics provides useful insight into difficult management situations in which the best efforts to solve a problem actually make it worse [2]. This approach encourages the inclusion of soft (as well as hard) factors in the model, *i.e.*, policy-related, procedural, administrative or cultural factors. To this end, in addition to its use in awareness training for management, system dynamics modeling may be of value to the identification and deterrence of insider threat.

Training on Recognition of Insider Threat Indicators

As we have argued, increased understanding and recognition of behavioral precursors is required to implement a system that anticipates or predicts potential insider attacks. We envision that, like the conventional staff development review process that is used by management to assess employee performance and manage career plans, a separate behavioral review and assessment process (with associated infrastructure and data repositories) is needed to support insider threat prediction and mitigation.

There is a critical need to provide training to managers and HR professionals to better prepare them to understand the risks of insider threats, to recognize possible behavioral/psychosocial precursors or indicators, and to train appropriate mitigation strategies. An evaluation instrument is straightforward to develop based on the indicators shown in Table 1, but effective use of the data by a predictive model such as that described in this chapter will depend on the consistent and appropriate use of such an instrument by managers and HR personnel. Misinterpretation of, or inconsistent judgments about these factors will have deleterious effects on the performance of the predictive model. The results of the preliminary empirical study described in the validation discussion (and shown in Figure 5) highlight the issue: without careful training, human experts will vary in their interpretation and application of possible behavioral indicators of insider threat.

To address the problem of raising awareness and facilitating consistent application and interpretation of insider threat predictive indicators, a new R&D program has been established by the Office of the Secretary of Defense (OSD) and executed through the Air Force Research Laboratory (AFRL) Human Effectiveness Directorate. The program seeks to develop game-based approaches to training and awareness of psychosocial indicators of insider threat that will accelerate learning of such concepts by managers or HR professionals.[11]

Mitigation

Once an organization has a greater understanding of the insider threat in its entirety, mitigating forces appear to reduce the likelihood of committing such acts or may defuse a specific threatening situation [31]. Highest on the list of mitigating factors is effective intervention by supervisors, co-workers, family members, and close friends. Intervention may lead to counseling, involvement with support groups, or medical assistance. It is essential, however, that those who might intervene recognize and respond to significant warning signs and symptoms. The hope is that organizations will be able to identify "at risk" individuals before or during employment to prevent compromise. Shaw and Fischer [31] suggest improvements for personnel management, which includes but is not limited to:

- Pre-employment screening, including personality traits, past and current behavior, and review of circumstances indicative of risk tailored to the vulnerable staff member. This necessitates Human Resources staff sensitization to behavioral traits and risk factors.
- Predisposition to warning behavioral characteristics should be mitigated through expectation setting to minimize potentially unmet expectations.
- Management recognition and facilitation of intervention techniques
- Improved awareness of changes in behavior, intervention and counseling techniques and approaches
- Documentation of problematic behavior and development of procedures to respond to such reports
- Organizational climate supportive of co-worker intervention—empowerment of employees to act and intervene on behalf of troubled co-workers through confrontation, counseling, Employee Assistance Program (EAP) referral and ability to report in confidence to security personnel.
- For introverted individuals, the creation of online environments designed to relieve work-related stress, *i.e.*, online EAPs or job-stress hotlines, electronic bulletin boards for logging anonymous complaints. These may fulfill a critical need not currently being met for staff exhibiting introversion.

[11] The research is conducted under SBIR program number OSD08-CR8 "Accelerated Learning through Serious Game Technology."

- Improved delivery and reinforcement of organizational rules of conduct and business ethics that the staff member can personally identify with—swift and public enforcement of violations of said rules of conduct.

In addition to the above-mentioned suggestions for human/personnel resource managers, a recently updated US-CERT report [7] has described best practices and recommendations based on CERT's analysis of over one hundred theft and fraud cases in which patterns not previously identified were revealed—theft or modification of information for financial gain and theft of information for business advantage. Recommendations, which should be included in management training to combat and mitigate insider threat, include these:

- Consider threats from insiders and business partners in enterprise-wide risk assessments.
- Clearly document and consistently enforce policies and controls.
- Institute periodic security awareness training for all employees.
- Monitor and respond to suspicious or disruptive behavior, beginning with the hiring process.
- Anticipate and manage negative workplace issues.
- Track and secure the physical environment.
- Implement strict password and account management policies and practices.
- Enforce separation of duties and least privilege.
- Consider insider threats in the software development life cycle.
- Use extra caution with system administrators and technical or privileged users.
- Implement system change controls.
- Log, monitor, and audit employee online actions.
- Use layered defense against remote attacks.
- Deactivate computer access following termination.
- Implement secure backup and recovery processes.
- Develop an insider incident response plan.

Although beyond the scope of this chapter, the authors would like to highlight that it is important for organizations to carefully consider whether their own internal management practices and examples might foster employee dissatisfaction, or unethical behavior, yielding a climate that leads to increased insider threats. The best, and first, line of defense is a commitment by the organization to ensure that insofar as it is possible, its employees are satisfied, engaged, and treated fairly.

6 Conclusions and Research Challenges

A major goal of this chapter is to motivate the combination of traditional cyber security audit data with psychosocial data, so as to move from an insider threat detection stance to one that allows prediction of potential insider presence. The hallmark of this recommended approach is to use organizational data in addition to cyber data

to support the analysis. The chapter described the challenges of this endeavor and progress in defining a usable set of predictive indicators, developing a framework for integrating the analysis of organizational and cyber data to yield predictions about possible insider exploits, and developing the knowledge base and reasoning capability of the system. We also discussed the types of errors that one expects in a predictive system versus a detection system, and how this can affect the usefulness of the results. We discussed challenges in evaluating predictive models for insider threats, and we summarized some preliminary studies that we have conducted to test performance of the models and their agreement with experts. A particular challenge that these studies revealed is that there is not high inter-rater agreement among HR experts regarding the influence of psychosocial factors; this may in part be due to varying interpretations of the meaning and severity of such behavioral indicators. This, and an associated need to develop and employ effective mitigation strategies, point to the need for more effective training programs for managers and HR personnel, as well as general employee awareness training, to raise awareness of and application of principles to address the insider threat.

Based on these considerations, we make the following high-level and broad recommendations for future research:

- **Prediction vs. Forensics**. The great research challenge is development of methods and tools for prediction to prevent or limit impact of insider exploits.
- **Transcending Intrusion Detection**. An increased focus should be on host-based insider detection and centralized situation awareness of distributed sensor data.
- **Exploiting Behavioral "Profiling."** There is a need for further research to further develop and update predictive behavioral indicators of malicious insider threats; the research need is focused on refinement of a taxonomy or characterization that captures insiders—behaviors, motives, methods, and psychological factors.
- **Testing and Evaluation**. A critical research need is more effective frameworks or methods for testing and evaluating the performance of predictive insider threat models. Major problems concern the lack of appropriate data sets, lack of ground truth, and challenges surrounding the acquisition and storage of test data because of organizational constraints.
- **Ethical and Privacy Issues**. The dialogue continues regarding tensions between security and privacy in the development and deployment of predictive insider threat monitoring systems, particularly when collection of behavioral/psychosocial data is involved.
- **Training and Awareness**. There is an important requirement to develop effective training, geared to general staff/employees, managers, HR professionals, and cyber security analysts for varying purposes, ranging from raising awareness of actions that may place individuals or organizations at risk to insider abuse, understanding of policies designed to combat insider threat, recognition of behavioral and cyber indicators that facilitate prediction of potential insider abuse, and appropriate mitigation strategies that are effective in defusing problems and that avoid exacerbating the situation.

7 Acknowledgments

The authors would like to acknowledge and thank the following people for their important contributions to the research reported in this chapter: Thomas Carroll, Thomas Edgar, Lyndsey Franklin, Ryan Hohimer, Lars Kangas, and Patrick Paulson. Report number PNNL-SA-67978.

References

1. Aleman-Meza, B., Burns, P., Eavenson, M., Palaniswami, D., Sheth, A.P.: An ontological approach to the document access problem of insider threat. In: Proceedigs of the IEEE International Conference on Intelligence and Security Informatics (ISI 2005), pp. 486–491 (2005)
2. Band, S.R., Cappelli, D., Fischer, L.F., Moore, A.P., Shaw, E.D., Trzeciak, R.F.: Comparing insider IT sabotage and espionage: A model-based analysis. Tech. rep., Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania, U.S.A. (2006)
3. Barbosa, R., Silva, N., Duraes, J., Madeira, H.: Verification and validation of (real time) COTS products using fault injection techniques. In: Proceedings of the Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems (ICCBSS '07), pp. 233–242. IEEE Computer Society, Washington, DC, USA (2007)
4. Brown, W.S.: Technology, workplace privacy and personhood. Journal of Business Ethics **15**(11), 1237–1248 (1996)
5. Butts, J.W., Mills, R.F., Baldwin, R.O.: Developing an insider threat model using functional decomposition. In: Proceedings of the Third International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security (MMM-ACNS 2005), pp. 412–417 (2005)
6. Cappelli, D.M., Desai, A.G., Moore, A.P., Shimeall, T.J., Weaver, E.A., Willke, B.J.: Management and education of the risk of insider threat (MERIT): Mitigating the risk of sabotage to employers? information, systems, or networks. Tech. rep., Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2006)
7. Cappelli, D.M., Moore, A.P., Trzeciak, R.F., Shimeall, T.J.: Common sense guide to prevention and detection of insider threats. Tech. rep., Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2009). 3rd edition, version 301. Available at http://www.cert.org/archive/pdf/CSG-V3.pdf.
8. Chinchani, R., Iyer, A., Ngo, H.Q., Upadhyaya, S.J.: Towards a theory of insider threat assessment. In: Proceedings of The International Conference on Dependable Systems and Networks (DSN 2005), pp. 108–117 (2005)
9. Costa, P.C.G., Laskey, K.B., Revankar, M., Mirza, S., Alghamdi, G., Barbar, D., Shakelford, T., Wright, E.J.: DTB project: A behavioral model for detecting insider threats. In: Proceedings of the 2005 International Conference on Intelligence Analysis. The Mitre Corporation (2005)
10. Doucette, P.J., Harvey, W.J., Hohimer, R.E., Martucci, L.M., Paulson, P.R., Petrie, G.M., Pike, B.A., Seedahmed, G.H.: Characterizing motion in video streams using supple knowledge. Tech. Rep. PNNL-16518, Pacific Northwest National Laboratory, Richland, Washington (2007)
11. Gabrielson, B., Goertzel, K.M., Hoenicke, B., Kleiner, D., Winograd, T.: The insider threat to information systems. State-of-the-art report. Tech. rep., Information Assurance Technology Analysis Center, Herndon, Virginia (2008)
12. Gelles, M.: Exploring the mind of the spy. In: Employees' Guide to Security Responsibilities. Texas A&M University Research Foundation, College Station, Texas (2005)

13. Greitzer, F.L., Frincke, D.A., Zabriskie, M.M.: Social/ethical issues in predictive insider threat monitoring. In: M.J. Dark (ed.) Information Assurance and Security Ethics in Complex Systems: Interdisciplinary Perspectives. IGI Global, Hershey, Pennsylvania (in press)
14. Greitzer, F.L., Moore, A.P., Cappelli, D.M., Andrews, D.H., Carroll, L.A., Hull, T.D.: Combating the insider cyber threat. IEEE Security and Privacy **6**, 61–64 (2008)
15. Greitzer, F.L., Paulson, P.R., Kangas, L.J., Edgar, T., Zabriskie, M.M., Franklin, L.R., Frincke, D.A.: Predictive modeling for insider threat mitigation. Tech. Rep. PNNL-SA-60737, Pacific Northwest National Laboratory, Richland, Washington (2008)
16. Infosec Research Council: Hard problem list (2005). Available from `http://www.infosec-research.org/docs_public/20051130-IRC-HPL-FINAL.pdf`. Accessed January 11, 2010.
17. Keeney, M., Kowalski, E., Cappelli, D.M., Moore, A.P., Shimeall, T.J., Rogers, S.: Insider threat study: Computer system sabotage in critical infrastructure sectors. Tech. rep., U.S. Secret Service and CERT Coordination Center, Washington, D.C., Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2005). Available from `http://www.secretservice.gov/ntac/its%5Freport%5F050516.pdf`. Accessed August 14, 2009
18. Kramer, L.A., Jr., R.J.H., Crawford, K.S.: Technological, social, and economic trends that are increasing u.s. vulnerability to insider espionage. Tech. Rep. 05-10, Personnel Security Research Center (PERSEREC), Monterey, California (2005)
19. Krofcheck, J.L., Gelles, M.G.: Behavioral Consultation in Personnel Security: Training and Reference Manual for Personnel Security Professionals. Yarrow Associates, Fairfax, Virginia (2005)
20. Lane, F.S.I.: The Naked Employee: How Technology is Compromising Workplace Privacy. American Management Association (AMACOM) (2003)
21. Magklaras, G.B., Furnell, S.M.: Insider threat prediction tool: Evaluating the probability of it misuse. Computers & Security **21**(1), 62–73 (2002)
22. Magklaras, G.B., Furnell, S.M.: A preliminary model of end user sophistication for insider threat prediction in it systems. Computers & Security **24**(5), 371–380 (2005)
23. Maybury, M., Chase, P., Cheikes, B., Brackney, D., Matzner, S., Hetherington, T., Wood, B., Sibley, C., Marin, J., Longstaff, T., Spitzner, L., Haile, J., Copeland, J., Lewandowski, S.: Analysis and detection of malicious insiders. In: Proceedings of the 2005 International Conference on Intelligence Analysis. The MITRE Corporation (2005)
24. Mayer, R.C., Davis, J.H., Schoorman, F.D.: An integrative model of organizational trust. Academy of Management Review **20**(3), 709–734 (1995)
25. Moore, A.P., Cappelli, D.M., Trzeciak, R.F.: The "big picture" of insider it sabotage across u.s. critical infrastructures. Tech. rep., Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania (2008)
26. Nardi, D., Brachman, R.J.: An introduction to description logics. In: F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, P.F. Patel-Schnieder (eds.) The Description Logic Handbook: Theory, Implementation, and Applications, pp. 5–44. Cambridge University Press, Cambridge, United Kingdom (2003)
27. Parker, D.B.: Fighting Computer Crime: A New Framework for Protecting Information. John Wiley & Sons, New York (1998)
28. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco, California (1988)
29. Rosenberg, R.S.: The workplace on the verge of the 21st century. Journal of Business Ethics **22**(1), 3–14 (1999)
30. Schultz, E.E.: A framework for understanding and predicting insider attacks. Computers & Security **21**(6), 526–531 (2002)
31. Shaw, E.D., Fischer, L.F.: Ten tales of betrayal: The threat to corporate infrastructure by information technology insiders analysis and observations. Tech. rep., Personnel Security Research Center (PERSEREC), Monterey, California (2005). Available from `http://handle.dtic.mil/100.2/ADA441293`. Accessed August 14, 2009.

32. Siegel, S.: Nonparametric Statistics for the Behavioral Sciences. McGraw-Hill, New York (1956)
33. Tabak, F., Smith, W.P.: Privacy and electronic monitoring in the workplace: A model of managerial cognition and relational trust development. Employee Responsibilities and Rights Journal **17**(3), 173–189 (2005)
34. US-CERT/CERT Coordination Center: 2004 e-crime watch survey—summary of findings. Tech. rep., U.S. Secret Service and CERT Coordination Center, Washington, D.C. Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2004). Available from http://www.cert.org/archive/pdf/ecrimesurvey05.pdf. Accessed January 11, 2010.
35. US-CERT/CERT Coordination Center: 2005 e-crime watch survey—survey results. Tech. rep., U.S. Secret Service and CERT Coordination Center, Washington, D.C. Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2005). Available from http://www.cert.org/archive/pdf/ecrimesurvey06.pdf. Accessed January 11, 2010.
36. US-CERT/CERT Coordination Center: 2006 e-crime watch survey—complete survey results. Tech. rep., U.S. Secret Service and CERT Coordination Center, Washington, D.C. Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2006). Available from http://www.cert.org/archive/pdf/ecrimesurvey06.pdf. Accessed January 11, 2010.
37. US-CERT/CERT Coordination Center: 2007 e-crime watch survey—complete survey results. Tech. rep., U.S. Secret Service and CERT Coordination Center, Washington, D.C. Carnegie Mellon Software Engineering Institute, Pittsburgh, Pennsylvania (2007). Available from http://www.cert.org/archive/pdf/ecrimesurvey07.pdf. Accessed January 11, 2010.
38. U.S. Department of Defense Office of the Inspector General (DoD): DoD management of information assurance efforts to protect automated information systems. Tech. Rep. 97-049, U.S. Department of Defense, Washington, D.C. (1997)
39. Wood, B.: An insider threat model for adversary simulation. In: Proceedings of the Research on Mitigating the Insider Threat on Information Systems. Arlington, Virginia (2000)
40. Zadeh, L.A.: Fuzzy sets. Information Control **8**(3), 338–353 (1965)

A Risk Management Approach to the "Insider Threat"

Matt Bishop, Sophie Engle, Deborah A. Frincke, Carrie Gates, Frank L. Greitzer, Sean Peisert, and Sean Whalen

Abstract Recent surveys indicate that the financial impact and operating losses due to insider intrusions are increasing. But these studies often disagree on what constitutes an "insider;" indeed, many define it only implicitly. In theory, appropriate selection of, and enforcement of, properly specified security policies should prevent legitimate users from abusing their access to computer systems, information, and other resources. However, even if policies could be expressed precisely, the natural mapping between the natural language expression of a security policy, and the expression of that policy in a form that can be implemented on a computer system or network, creates gaps in enforcement. This paper defines "insider" precisely, in terms of these gaps, and explores an access-based model for analyzing threats that include those usually termed "insider threats." This model enables an organization to order its resources based on the business value for that resource and of the information it contains. By identifying those users with access to high-value resources,

Matt Bishop
Dept. of Computer Science, University of California at Davis e-mail: bishop@cs.ucdavis.edu

Sophie Engle
Dept. of Computer Science, University of California at Davis e-mail: sjengle@ucdavis.edu

Deborah A. Frincke
Pacific Northwest National Laboratory e-mail: deborah.frincke@pnl.gov

Carrie Gates
CA Labs., Inc. e-mail: carrie.gates@ca.com

Frank L. Greitzer
Pacific Northwest National Laboratory e-mail: frank.greitzer@pnl.gov

Sean Peisert
Dept. of Computer Science, University of California at Davis e-mail: peisert@cs.ucdavis.edu

Sean Whalen
Dept. of Computer Science, University of California at Davis e-mail: shwhalen@ucdavis.edu

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_6,

we obtain an ordered list of users who can cause the greatest amount of damage. Concurrently with this, we examine psychological indicators in order to determine which users are at the greatest risk of acting inappropriately. We conclude by examining how to merge this model with one of forensic logging and auditing.

1 Introduction

Modern culture considers the "insider" a term of both honor and opprobrium, implying that the person so identified is privy to access or information that others are excluded from. Therefore, people should regard their words or actions as less questionable because they are undoubtedly based on more information, or better information, than others have available. But the insider, being privy to that access or information, can wreak far more damage than an outsider by exploiting their knowledge and abilities.

These widely held opinions define "insider" as some mixture of access and knowledge—and more often than not, the term is never explicitly defined. Even when it is, the definitions are often contradictory or specific to a particular environment. For example:

1. An insider is someone who is authorized to use computers and networks [39];
2. An insider has access to the keying materials or full control of some nodes [30];
3. An insider has "access, privilege, or knowledge of information systems and services" ([9], p. 10)
4. An insider is anyone who operated inside the security perimeter [32]; and
5. An insider is a database subject who has personal knowledge of information stored in one or more fields marked confidential [18].

These definitions are different. For example, the second definition requires "full control" of nodes, whereas the fourth simply requires someone to be within the security perimeter. The first applies to anyone who has permission to use the computers and networks, whether or not they have full control of nodes or are within the security perimeter. The third definition does not require authorization; it merely requires knowledge. And the last definition is specific to database systems.

Two factors underlie these definitions. The first is *access*. All require the insider to have some degree of access to resources. The mode of access differs: "use," "read," "control," or "write," for example. So does the level of access, for example "privileged," "ordinary," or "full." The second is *knowledge*. Knowing something about the information systems and services or values in the database implies the ability to act on that knowledge. For our purposes, we consider knowledge a form of access because it implies one knows data about *that particular organization*. In other words, knowing that a user named "root" has password "gleep" is useless unless one knows that the Department of Redundancy Department has on its server that particular user with that particular password.

A third factor, implicitly mentioned by the above, is *trust*. Authorization implies some level of trust, as does access to keying materials or operating within a security perimeter. As with knowledge, we consider trust a form of access because in practice, organizations trust those with access. How this trust is granted differs: employment implies some level of trust; a security clearance implies a (perhaps more general) type of trust. For our purposes, trust as an intangible expression of assurance in the rectitude or predictability of an individual is not relevant. If that trust leads to the granting of access, or the imparting of knowledge, then it is relevant because of the effects of the trust, not because of the trust in and of itself. This again causes us to focus on access [7].

The implication of the distinction between "insider" and "outsider" is that they have different capabilities or affect the organizations in question differently, and that one can provide a precise characterization of both the differences between an "insider" and an "outsider." The problem is that such a characterization is rarely attempted, the papers relying on the reader's intuition; and when it is attempted, the characterization is either imprecise or specific to a particular set of facts. The above examples show this. Further, the difference in capabilities relies on knowledge or access, and does not provide a precise set of knowledge or access capabilities that lead to the distinction between "insider" and "outsider."

Our theme is that the distinction between "insider" and "outsider" is not binary; rather, there are "attackers" with varying degrees and types of access. One can call some set of these attackers "insiders," with the complement being the "outsiders," but countermeasures should focus on the access and not on whether the attackers are insiders. Thus, we see attacks as spanning a continuum of levels and types of access, and use that as the basis of our discussion. We emphasize that people comfortable thinking in terms of "insiders" and "outsiders" can superimpose that partition on our notion of "attackers with varying levels of access." That partition, however, will vary based on circumstances and environment.

The next section describes our model in detail, discussing how we use access to identify sets of users and resources of interest. We can then refine membership in our sets using various techniques; here, we focus on psychological indicator-based assessment techniques. Following this, we examine countermeasures.

2 Insider Threat Assessment

Methods of detecting insiders have two distinct parts. First is to determine whom to worry about; in other words, who has the capability to launch such an attack. Second is to determine who is likely to attack; this is psychological, because having a capability does not mean it will be used. We consider these separately.

As with all security, we begin with the security policy. This poses an immediate problem. *Which* security policy do we begin with? The obvious answer (the current policy) is insufficient because policies are imprecise. Even policies expressed using policy specification languages only capture part of the site's intended policy. In fact,

a "policy" in practice is an abstract expression of requirements, desires, and other factors. As it is refined, different instantiations emerge.

Carlson's Unifying Policy Hierarchy [12, 5] provides a framework for capturing the different refinements (see Table 1). At the top of the hierarchy is the *Oracle Policy*. This is a mechanism that can supply a correct, precise answer for any policy question asked. It is a management mechanism, not a technical one. It can include non-technical factors such as intent, custom, law, and so forth. In practice, it is non-deterministic and not Turing-computable. Hence it should be viewed as the "ideal, intended policy" even when that policy, or parts of that policy, are unclear. An example statement in an Oracle Policy would be:

> Mary is allowed to read patient medical data for the purpose of compiling statistics for CDC reports on the geographic spread of medical conditions.

This statement specified an action ("read"), a resource ("patient medical data"), and a condition ("for the purpose of compiling statistics ..."). The purpose embodies intended use; Mary cannot read the data, for example, to sell it to anyone. In terms of the domains in Table 1, the intent is a condition e that must be met for the subject s (Mary) to be able to perform action a (read) on object o (the patient medical data).

The Oracle Policy contains elements that are ambiguous when mapped onto a computer system; it also contains elements that are infeasible for a computer system to enforce. For example, what exactly is "patient medical data?" Presumably, data on the system is labeled in some way, for example by being in certain (sets of) directories or having a tag marking it as PATIENT or CONFIDENTIAL. But if the data is stored in a different file, and unlabeled, Mary may not realize it is confidential, protected data and thus she may reveal it. This is an imprecision. To take this a step farther, Mary has authority to access even data that is so labeled when she needs to for her job (compiling statistics). She does not have that permission when she needs to access it for a purpose unrelated to her job (selling the data). This is embodied in the condition e, intent, mentioned earlier. But the computer system and controls

Table 1 The Unifying Policy Hierarchy. The entities are subjects $s \in S$, objects $o \in O$, and actions $a \in A$. The condition $e \in E$ describes additional constraints on the ability of s to execute a on o due to external factors such as intent. The "Run-Time Instantiation" is, strictly speaking, not a policy, but instead a description of what a user can do, whether those actions are authorized or unauthorized; that is, it encompassed unauthorized actions possible due to security flaws.

Level of Policy	*Domain*	*Description*
Oracle Policy	$S \times O \times A \times E$	What should ideally be authorized, including intentions.
Feasible Policy	A subset of $S \times O \times A$ containing system-definable entities	What can be authorized in practice, considering system constraints.
Configured Policy	A subset of $S \times O \times A$ containing system-defined entities	What is allowed by the system configuration.
Run-Time Instantiation	A subset of $S \times O \times A$ containing system-defined entities	What is possible on the system, factoring in any flaws or vulnerabilities.

cannot read minds or divine intent with perfect accuracy any more than humans can; thus, security controls in this situation do not impede access.[1] So, those aspects of the policy that deal with intent, for example, are infeasible for a computer system to enforce.

Thus, the Oracle Policy is an idealized statement of what the exact security policy would be. It is unambiguous, and able to provide a decision about each quadruple. It partitions all possible states into two sets, "allowed" and "not allowed." The Oracle Policy may evolve over time; but in all cases, at all times, it can determine whether a given state is in the "allowed" partition of states, or the "not allowed" partition of states.

The *Feasible Policy* is a refinement of the Oracle Policy that can actually be implemented on a (possibly idealized) computer system. It differs from the Oracle Policy in that the Feasible Policy is grounded in technology and feasible procedures. For example, the above Oracle Policy statement would be represented as

> The account "mary" has read access to patient medical data.

Here, the notion of intent has been jettisoned, because (as of this writing) a computer system cannot determine intent. The exact mechanism that the computer uses to identify data as "patient medical data" is left unspecified because there are several ways to do so; all are implementation-dependent.

In most cases, the Feasible Policy does not capture the Oracle Policy precisely. Thus, there are "gaps" between the Feasible Policy and the Oracle Policy. For example, under the Feasible Policy, Mary is allowed to read patient data even when she plans to sell it. Under the Oracle Policy, Mary would not be allowed to read the data in that case.

In fact, the above statement glosses over a second gap: the difference between Mary and her account. Anyone who gains access to Mary's account can act as Mary. Therefore, the Feasible Policy does not restrict access to Mary. Rather, it restricts access to the associated account "mary."

The instantiation of the Feasible Policy on a particular system is called the *Configuration Policy*. Unlike the Feasible Policy, the Configuration Policy is aimed at a particular system, and its features constrain the instantiation of the policy. For example, the Configured Policy might represent the above Feasible Policy statement as:

> The account "mary" has read access to files in the directory "DBMS:patientdata" with suffix "pmd"

because on this system, all patient data is kept in files with names ending in "pmd" and in the directory "DBMS:patientdata." This expression can be instantiated using the file access controls on the system, the naming conventions for files and directories, and the directory structure. As with the Oracle and Feasible Policies, there are gaps between the Feasible Policy and the Configuration Policy. For example, if patient medical data were put in files other than those identified in the

[1] Instead, they try to detect when this type of breach has occurred, or try to deter this type of breach through a variety of means.

Configuration Policy, Mary might not have access to them. Thus, this denies her access to data that the Feasible Policy specifies she should have access to.

This also points out an interesting aspect of the gaps. The gap between the Oracle and Feasible Policies identified above is one of *granting* rights so that the Feasible Policy does not contain a restriction (denial of rights) that the Oracle Policy imposed. The instantiation provides subjects more rights than they should have. The gap between the Feasible and Configuration Policies identified above is one of *denying* rights, so that the Configuration Policy grants a right that the Feasible Policy does not contain. The instantiation provides subjects with fewer rights than they should have. For our purposes, we focus on the former type of gap. Gaps created by the deletion of rights enable denial of service attacks, and they can be treated in the same way as gaps created by the granting of rights.

Ideally, when the Configuration Policy conforms to the Feasible Policy, the Configuration Policy would describe the capabilities of every subject on a system. Were the system implemented correctly, the subjects would be so constrained. But in practice, software has security flaws, called *vulnerabilities*. These vulnerabilities add rights.[2] For example, suppose a buffer overflow in a privileged program allows Mary to obtain the privileges of a system administrator. The rights she has are no longer those that the Configured Policy gives her. Therefore, the *Run-Time Instantiation* describes the actions that subjects can take on objects.

As with the other pairs of policies, there exist gaps between the Configured Policy and the Run-Time Instantiation. In our example, when Mary exploits the vulnerability in the privileged program to acquire the system administrator privileges, she can now access files that the Configured Policy intended to deny her access to.

2.1 Example

Consider a company that has a policy of deleting accounts of employees who leave (either voluntarily or because they are fired). The system administrators are responsible for doing this. Thus, there are tools to delete accounts, and a mechanism for determining when an account is to be deleted–perhaps the Human Resources Division sends the system administrator a note. The three highest policies would be:

- Oracle Policy: When an employee leaves, the employee is to lose all access to the company systems.
- Feasible Policy: Upon notification from the Human Resources Division that an employee has left the company, the primary account associated with that employee is to be disabled.
- Configuration Policy: The password and remote access authorization mechanisms for the primary account associated with an employee are to be disabled, so the employee cannot access the account.

[2] Of course, they may also delete rights.

Here, the Oracle and Feasible policies overlap considerably, but two gaps are apparent. The first stems from the relationship between "an employee leaves" and "notification ... that an employee has left." If the employee does not notify the company she is leaving, but simply stops coming to work, the company may not realize that the employee has left. Thus, the account will not be deleted even though the employee has left. A second, less speculative version of this gap arises when the Human Resources Division fails to notify the system administrators in a timely fashion. In that case, the departed employee still has access to the company systems.

The second gap is the distinction between "lose all access to the company systems" and "the primary account ... is to be disabled." The assumption here is that without an account, the departed employee cannot access the system. This assumption relies on the employee having no access to any other account, or any other means for gaining access (such as connecting to a web server on a company system or using FTP to access files in an FTP directory). Should the employee be able to do so, the Feasible Policy will grant rights that the Oracle Policy denies, creating the gap.

A similar gap exists between the Feasible Policy and the Configured Policy. The Feasible Policy requires that the employee be denied access to the primary account. The Configured Policy describes how this is to be done on the system in question—here, by disabling the password (so the user cannot authenticate correctly by supplying the password) and by deleting any indicators of "remote trust" that enable remote users to access the account without authenticating locally.[3] The gap lies in the assumption that those actions will disable the account. If the user has set up a program that runs every evening and mails important data to the employee's Gmail account, then the steps required by the Configuration Policy do not achieve the desired goal, that of disabling the account.

Finally, even were there no gaps between any of the above three policies, one must consider the actual policy enforced by the system, the Run-Time Instantiation. Suppose the system runs a web server with a buffer overflow vulnerability that starts a command interpreter when an attacker triggers the overflow. Thus, the Run-Time Instantiation gives the departed employee access to the company system.

This suggests a precise definition for the notion of an "insider."

Definition 0.1. Let P_L and P_H be representations of a policy at different levels of the Unifying Policy Hierarchy. If a subject has a different set of rights in P_L than it has in P_H, it is called an *insider*. An *insider attack* occurs when an insider employs any rights that exist in P_L and that do not exist in P_H.

In our example above, suppose Nancy leaves the Marvelous Company for a better job. The Human Resources Division of the Marvelous Company, in accordance with its policy, directs the system administration to disable Nancy's account. But the system administrator is ill that day, and does not do so. Then Nancy still has the same trusted access to the company systems that she had when she was an employee. This is a classic case of an "insider."

[3] For example, on a UNIX or Linux system, the hosts.equiv, .rhosts, and .shosts files would be changed appropriately or deleted.

More generally, the introduction identified three key properties that most definitions of the term "insider" are based on:

- *Access*: The insider needs some degree of access to resources. In the above definition, the subject has rights to certain resources. Those rights give it some form of access. Thus the definition covers this property.
- *Knowledge*: The insider needs to know about the resources available to it. A subject (presumably) knows it has a particular right, and what it does; hence, it knows about the resource involved. Thus, the definition covers this property.
- *Trust*: The insider must be trusted to honor the restrictions imposed on it. In the definition, P_L provides the subject with rights that it could use to exceed restrictions that P_H poses on it. The subject is trusted not to use those rights. Hence the definition covers this property.

Compare these to our definition of "insider":

- *Access*: A subject cannot employ a right without access, because if there is no access, any rights are effectively inert and cannot be used. Thus the definition covers this property.
- *Knowledge*: When a subject employs a right, the subject must know that it has the right, and must know how to use it. Thus the definition covers this property.
- *Trust*: A subject trusted not to use rights that P_L gives it, but that P_H does not, uses those rights. Hence this definition describes the betrayal of trust underlying an insider attack.

2.2 Summary

The definition of "insider" presented above is based on the exploitation of gaps in the representation of a policy at different levels. It says nothing about which insiders to fear, because most people who meet the definition of "insider" pose little to no threats. So, we first determine who poses risks by establishing the sets of users about whom we must be concerned. Once this access-based assessment is complete, we turn to an assessment of the psychological profiles of people who might try to exploit these gaps, and thereby launch insider attacks. For convenience, we refer to these people as "malicious insiders."

3 Access-Based Assessment

Given the above definition of insider, we now examine how to classify entities (subjects, usually people but possibly including autonomous agents) and resources (objects) in order to determine the risk of an insider attack, and the exposure that would result from such an attack.

We build on our observation that the differences in access granted by different layers define the insider. As an example, recall that the Oracle Policy gave Mary access to personal medical data under some conditions, but the Feasible Policy tied that access to the representation of Mary on the system, namely Mary's account. Thus, we describe a model that characterizes subjects in terms of access to resources.

Our model, the *Attribute-Based Group Access Control Model (ABGAC)* [5, 6], groups both subjects and resources into sets defined by attributes of interest. Role-Based Access Control [17] is a specialization of this model, in which the subjects' attributes of interest are the job functions of the subject. To illustrate the difference, consider system administrators, a well-defined role for RBAC at most institutions. The access may differ based on time of day, which—to use RBAC—must be folded into the job function. With ABGAC, we simply define the groups by two attributes: system administration (also a role) and being in the building from midnight to 8:00AM.

We define the following components of the model.

Definition 0.2. A *resource pair* consists of an object and an access right.

For example, the pair (personal medical record, read) is a resource pair describing how some set of subjects can access the personal medical record. Similarly, (building, enter after midnight) describes the ability to enter a building after midnight. The objects and rights may be any objects and rights of interest; they need not be virtual resources, or exist on computers.

Often, a set of different resource pairs will be similar enough to be treated the same way. An example would be a set of printers to which jobs are assigned based on load. If a user can print a document on one printer, she can print the document on any of them. The next definition captures this.

Definition 0.3. A *resource domain* is a set of resource pairs.

The utility of this definition is actually greater than the above paragraph suggests. Specifically, an important characteristic of resource domains is that they are oriented towards the object and not the subject. For example, the ability to print on a printer may enable a covert channel: send the file to be printed, and see if it prints immediately. If so, that corresponds to a "1" bit. If it is queued for printing later, that corresponds to a "0" bit. In this case, an appropriate resource domain would consist of two resource pairs, one for printing a document on the printer (thereby manipulating the covert channel) and one for determining whether a specific document was printing or waiting to print (reading the covert channel).

In order to launch an attack, a malicious insider will often need access to multiple resource domains. For example, the attacker might need to read information from the resources containing personal medical information, and then save it in a local file for later reference (or transmission). Assuming the pairs (database entry of personal medical information, read) and (local file, write) are in two different resource domains, the following definition ties them together in this situation:

Definition 0.4. An *rd-group* is the union of resource domains.

The rd-groups combine with subjects' protection domains to define groups of users. Intuitively, associated with each rd-group is that set of users who can access all the resources in the rd-group in the manner indicated by the resource pair in that rd-group. Thus,

Definition 0.5. A *user group* is the set of all subjects whose protection domains contain the associated rd-group.

As an example of how these definitions fit together, consider an organization's information technology group. Among the resources it manages are desktops printers. The organization's operations rely on customer addresses, customer credit card information, and customer purchasing history (collectively called "customer data"). The company also tracks its CEO's email for legal reasons. There are two senior system administrators named Alice and Bob, and two junior system administrators, named Charlie and Eve. All system administrators have access to the customer data, but only the senior sysadmins have access to the financial information and CEO's email.

Our challenge is to find the insiders associated with the CEO's email.

From the above information, we define resource pairs based on access to the resources. The resources are the CEO's email, customer data, desktops, and printers. These can be accessed for reading, writing, or physically (as, for example, when Eve takes a printout off the printer). Thus, the resource pairs are:

(customer data, read)	(desktop, read)		(CEO email, read)
(customer data, write)	(desktop, write)	(printer, write)	(CEO email, write)
	(desktop, physical)	(printer, physical)	

Next, we define the resource domains of interest. rd_1 captures the ability to read the CEO's email, necessary for an attacker to acquire it. rd_2 and rd_3 represent the ability to save the email to a resource that can then be removed from the organization's premises.

rd_1 = { (CEO email, read) }
rd_2 = { (desktop, write), (desktop, physical) }
rd_3 = { (printer, write), (printer, physical) }

For an attacker to acquire the CEO's email, the attacker must somehow read it and then get a physical copy of it. Thus, the rd-groups of interest are:

$rdg_1 = rd_1 \cup rd_2$ = { (CEO email, read), (desktop, write), (desktop, physical) }
$rdg_2 = rd_1 \cup rd_3$ = { (CEO email, read), (printer, write), (printer, physical) }

We next look at the user groups induced by rdg_1 and rdg_2. As only Alice and Bob have read access to the CEO's email, and all system administrators have both write and physical access to the printers and desktops, then rdg_1 and rdg_2 are clearly subsets of the protection domains of Alice and Bob. Thus, the relevant user group is composed of Alice and Bob.

At this point, the Unifying Policy Hierarchy must be considered. Throughout this example, "Alice" and "Bob" are people, but as noted earlier computer systems represent people with accounts. Therefore, anyone with access to Alice's or Bob's account also poses a threat. This illustrates a critical aspect of this model.

When one speaks of a "resource," one must identify all types of access to that resource. For physical resources such as printers, this is usually straightforward.[4] For virtual resources, such as the CEO's email, access may be obtained not only in the usual ways (such as by reading the files using the appropriate system calls), but also in less usual ways (such as reading the disk device directly and reassembling the file from the information on the disk, or reading temporary buffers in memory). Ideally, all these methods will be identified and added to the set of resource pairs.

A similar consideration holds for Alice and Bob; without loss of generality, use Alice as an example. If she has a home computer and uses that to access the company systems, it is likely that she occasionally walks away from the computer (to use the bathroom, say) without locking it or logging herself out. This gives access to her account to anyone in the house. When she takes it to a repair shop (because the company will not repair personally owned systems), the repair people may obtain her company password or be able to acquire it through nefarious means. All these entities must be considered when user groups are developed.

This leads to the question of risk. Clearly, there are too many possibilities to enable the analysis of all of them. So, some type of risk analysis mechanism must winnow out those possibilities for which the danger of attack is small.

Our approach is to determine the cost of a successful insider attack. The analyst can then determine whether the benefit of not defending against such an attack outweighs the cost of defending, taking into account the likelihood of the attack.

For purposes of discussion, let U be the set of user groups and D the set of rd-groups. Define the cost function $C\colon U \times D \to \mathbb{R}^n$, where $\mathbb{R}^n$ is the set of vectors describing the costs of compromise. This suggests two approaches.

The first approach is to minimize the impact of a successful attack. As C induces a partial order over the elements of its range, one can minimize the vector components of the value of C for any element of its domain. It may not be possible to minimize all of them simultaneously. In that case, management must decide which values are most critical to minimize.

The second approach is to minimize the number of subjects who pose a threat. This approach is appropriate when the costs of compromise are high enough that they must be defended against, yet only post hoc procedures will work. Returning to our example of personal medical information, Mary simply must have access to it to do her statistical analysis. Denying her that access means the company will not obtain critical information. Hence the cost of compromise cannot be minimized, because the personal medical information either leaks or it does not leak. Effectively, cost of compromise is a delta function, taking on the values 0 and the cost of

[4] But not always. Consider that a single printer may be virtualized on a computer, in which case all aspects of access must be considered. Similarly, if the printer is mirrored so whatever it prints is also saved to a disk or printed elsewhere, access to those other resources is equivalent to access to the printer in question.

compromise. But if the number of users who can access the personal medical information is minimized, then the number of people who could leak the information is also minimized. This may reduce the probability of the information being leaked, rather than the cost of leaking it.

The Unifying Policy Hierarchy model poses a challenge, because determining the number of subjects with access to the data requires an application of that model. An example for Alice was given above. Hence there will be unknown subjects with access to the data. The approach we take is to treat the known users as proxies for unknown users. Suppose Alice's son uses the home personal computer to visit a web site and accidentally download a malicious applet, which installs a keyboard sniffer. After his bedtime, Alice uses the home personal computer to connect to the company's computer, and the author of the malicious applet gets the company computer's network address, Alice's account name, and Alice's password. He then accesses Alice's account. Even though he is an unknown subject, the analyst can approximate the effect of his compromise by examining the effect of Alice's compromising the data.

This takes us to an examination of people: how should the analysis determine whether an individual is likely to be a malicious insider, or to give access accidentally to a malicious insider?

4 Psychological Indicator-Based Assessment

Research characterizing psychological profiles of malicious insiders focuses largely on case studies and interviews of individuals convicted of espionage or sabotage [19, 25, 31, 16]. Band *et al.* [2] and Moore *et al.* [29] summarize findings that reveal behaviors, motivations, and personality disorders associated with insider crimes such as antisocial or narcissistic personality. Anecdotal research is post hoc, mostly derived from interviews with convicted criminals, and speculative in its predictive value. Also, assessing such personality disorders and motivations in an organization is difficult at best, and management or human resources staff may not be able to do so accurately and consistently because a typical organization does not administer psychological or personality inventory tests. Another challenge is that no studies assess and compare the prevalence of these "insider threat" predispositions with occurrence rates in the overall employee population—an important comparison needed to validate the hypothesized relationship.

Nevertheless, the body of research using case studies warrants continued efforts to address psychosocial factors. One approach is to develop predictive models that correlate the psychological profiles or behaviors that have been observed in case studies to insider crime–for example, personal predispositions that relate "... to maladaptive reactions to stress, financial and personal needs leading to personal conflicts and rule violations, chronic disgruntlement, strong reactions to organizational sanctions, concealment of rule violations, and a propensity for escalation during work-related conflicts" ([2], p. 15 and Appendix G). While the factors described in

the extant research reflect psychological profiles inferred from case studies and interviews by staff psychologists, an alternate approach would attempt to synthesize a set of indicators from this research and derive a set of corresponding observables that would serve as proxies for the indicators. These observables could then be extracted from a manager's evaluations of staff behavior and performance. A complementary approach is to develop instructional methods to raise managers' awareness of, and enhance their ability to detect, the warning signs of potential insider attacks. Examples of this approach are the workshops and interactive training that US-CERT [29] offers, and a research and development program at the Office of the Secretary of Defense that is developing game-based methodologies to be used in this approach [1].

Greitzer, Frincke and Zabriskie [21] discuss the availability and appropriateness of different types of behavioral/psychosocial data for detecting malicious, or potentially malicious, insiders. While they conclude that certain types of data monitoring would be inappropriate for reasons of privacy and ethics, they find that several sources of employee data are worthy of consideration, as summarized below.

- *Personal Information*. Generally, use of personal information within federal institutions is not likely to be appropriate or legal, no matter how useful it might be in mitigating insider threats. The employee's legal right to, and expectation of, privacy for medical records and life events such as birth, adoption, or divorce trumps the organization's desire to predict insider attacks. An employee's marital and financial problems likely could not be used either. However, such life events are known to increase stress in many individuals; signs of trouble may arise not only from such personal events as divorce or death in family, but also from work-related stress due to performance issues [2].
- *Manager's assessment of employee morale*. An attentive manager should be mindful of an employee's personal situation and whether that employee's behavior reflects stress or other issues. Such attentiveness creates a supportive working environment that leads to higher employee satisfaction and less likelihood of disengagement, stress, and resulting insider attacks. Therefore, regardless of the personal life events that may underlie behavior, an attentive manager can provide judgments useful in a monitoring and analysis program. Further, an auditable trail of such information lets employees examine and correct any biased opinions as well as protecting the organization from liability.
- *Social and Organizational Information*. Unlike personal information, most work-related employee data may be used legally to observe, report, and correct inappropriate or suspicious behavior. Many employees receive annual performance evaluations that may address issues about productivity, attitude, and interpersonal skills. Recurring "rule conflicts" or "personality problems" may be observed before actual insider threat events. These observations might be elements of strategies to reduce the threat of insider attacks. Many authorities suggest that managers should keep detailed records and note trends about events that result in employee disciplinary action [2]. Feedback obtained from "360 degree evaluations" by associates, direct reports, and managers should be useful in assessing psychosocial factors (particularly if a manager is reluctant to provide negative

feedback). Also, employee records may contain complaints by or against the employee and information related to employment applications such as education and work history.

A predictive approach enables an attentive manager to speak with stressed employees and address underlying problems in order to avert an insider attack. But a predictive approach risks potential damage that may arise from a false accusation. Adopting a predictive approach requires distinguishing between detecting *indicators* that precede a crime and detecting criminal *evidence*. In a predictive model, detection involves identifying precursors, not identifying the actual exploit. Indicators may be misleading due to the uncertainties of behavioral measures. Therefore, it is critically important to keep the human in the loop; a predictive system should be a tool for "tapping analysts on the shoulder" to suggest possible "persons of interest" on whom to focus limited resources. The underlying concept of the system is to preserve the analyst's ability to make key decisions, and responsibility for those decisions, while helping to reduce the information load, the risk, and the cost of false alarms.

Greitzer and his colleagues [21, 22] and Greitzer and Frincke (Chapter TBD, in this volume) describe a predictive modeling approach that combines traditional cyber security audit data with psychosocial data, to support a move from an insider threat detection stance to one that enables prediction of potential insider presence. Based on case studies that have been reported in the literature, this approach holds that the objective of predicting or anticipating potential insider threat risks is best served by using organizational/behavioral data in addition to cyber data to support the analysis. Without incorporating psychosocial data, the authors argue that prediction is extremely difficult because analysis of cyber data alone is likely to reveal malicious activity only after it has occurred.

A major focus of the predictive analysis approach is the specification and incorporation of psychosocial factors, and description or delineation of observable proxies for such factors. Greitzer and Frincke (see Table 1 in Chapter TBD, this volume) list a set of twelve such proxies, referred to as psychosocial indicators and describe a study designed to test the model against expert judgments of severity of different combinations of indicators. A particular aspect of the model development and expert knowledge acquisition that is particularly relevant to the current chapter is the nature of risk, as interpreted by the subject matter experts consulted in developing the model. Figure 1 illustrates the indicators that we identify in the model that contribute to the potential risk of an individual to become an insider threat.

An important assumption is that any such indicator is worthy of consideration *only* if the employee exhibits extremely serious or grave manifestations of the indicator. Moreover, based on interviews conducted with a limited set of human resources experts, Greitzer and colleagues concluded that these twelve factors have varying levels of strength or utility in determining the risk of insider attack. The five factors shown in the darkest shade (disgruntled, accepting feedback, anger management, disengagement, and disregard for authority) are generally considered to be more serious than the other factors. As a general rule, it would take a preponder-

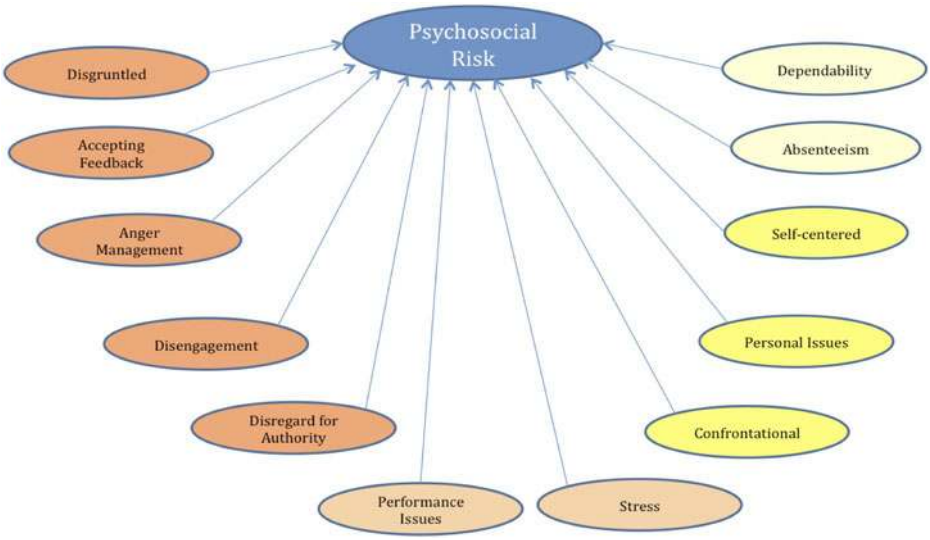

Fig. 1 Indicators that an individual is a potentially malicious insider. The darkness of each ellipse reflects the seriousness of the indicator.

ance of the other (lesser) indicators to lead one to be concerned about an individual employee, compared to perhaps only one or two of the most serious indicators.

In addition, the research conducted to date has found a fair degree of variability in the judgments of human resources experts with regard to the association of these indicators with potential risk of insider attack. Initial studies performed by Greitzer and Kangas [20] asked human resources experts to assign threat or risk values using a scale numbered from 0 (least) to 10 (greatest) of various combinations of the twelve indicators, ranging from only one indicator observed to between 4 and 6 indicators observed. Only a subset of possible combinations of indicators was practical in these studies because of the large number of possible combinations. It was clear that raters were not simply counting the number of indicators in arriving at their risk assessment (because there was no significant relationship between the risk values and the number of indicators identified as present). However, while there was some degree of consistency among human resource raters (with R^2 in the range of 0.4–0.5 in general), there was not uniform agreement. Some raters might have been interpreting some indicators differently than other raters. The nature of the rating task is very demanding and possibly a source of "information overload" in its own right, particularly given the number of cases that were included in the study. A different approach to this knowledge engineering task, in which cases are presented more as narrative descriptions of the identified indicators rather than the indicator labels merely being checked off, is being studied.

Another challenge in modeling the combination of psychosocial indicators and workstation and other cyber indicators is the large difference in data volume and

"tempo" between psychosocial and cyber indicators. Clearly the volume of data for workstation and cyber indicators is massive, and the volume of psychosocial data is relatively sparse. This in itself is not a great challenge, but temporal differences do present difficulties for predictive models that reason about risks based on integration of cyber and psychosocial data. Of most concern is the fact that these different types of data are *asynchronous*. Cyber data is monitored in real time (although not necessarily analyzed and available immediately) and are generally available in chronological order. In contrast, psychosocial data is collected infrequently and may not be in any sort of chronological order. For example, an employee's suspicious computer activity may be monitored at a given point in time, and at some later time it might be learned that the employee has been disgruntled. At the time the suspicious activity was analyzed, that activity alone might have been insufficient to yield a risk value significant enough to justify any security action. When the late information about disgruntlement is learned, the analysis system must be able to integrate it with the threat analysis. Asynchrony in such data and analysis imposes constraints on how data is maintained, how and how long analyses are conducted, and on the way components of the predictive model share information and analysis results. A sophisticated reasoning mechanism and information architecture is required to enable such asynchronous assessments to occur.

5 Application of Risk to System Countermeasures

In this chapter, we have discussed two constructions for defining insiders: our modified version of the Unifying Policy Hierarchy, as well as the ABGAC framework. These constructions offer a more precise means of defining insiders with respect to *insiderness* than has been previously captured. The framework and hierarchy also provide a means for defining first-order protection in the form of access control.

However, defining access control rules for insiders—even more precise ones—still leads to a conflict: how do we mitigate damage (either accidental or intentional) while allowing these users to do their jobs? The problem is that a situation may arise that the developers did not, and could not, foresee. For example, consider the "doomsday scenarios" posited during the cold war in novels such as *Fail-Safe* [11] and movies such as *Dr. Strangelove* [26], in which a control center is trying to recall bombers sent to attack the Soviet Union by accident. The aircraft either ignore what the commander has been taught is a spurious transmission from the enemy (even though, in this case, it is not spurious and from the command center), or they do not receive the recall code because a missile has destroyed the radio receiver.

These difficulties arise because security systems are relatively rigid—they serve as firewalls that allow permitted actions and block forbidden actions. This is not the way firewalls work in human society. For example, security in an industrial building is provided by security guards, access control cards, security cameras, and logging of door accesses. One security policy might be that only authorized people can enter a development lab. But in practice, doors can be propped open and access

cards can be "borrowed." So, the building should have a security guard who can notice someone acting unusually—like carrying an expensive piece of equipment—and ask them for identification. A laptop being disconnected from the Internet can be detected by examining wireless access point logs and then looking at the history recorded by nearby security cameras. By more tightly interlocking the components of system security (authentication, authorization, and auditing), and by accepting that the system may be in a somewhat uncertain state, the organization runs more smoothly and efficiently without taking on undue risk.

Ideally, a system should automatically block all forbidden actions and permit all allowed actions using some variety of access control matrix [27]. But Jones and Lipton showed that this is infeasible [23]. For example, suppose that a defense involving physical access is broached: a laptop owner walks out of her office for a few minutes, and leaves her office unlocked. If she did not lock her screen, then the failure of the physical access protection defeats the protection by authentication—someone can walk in and start using the laptop. One can try to protect against this by using an anomalous behavior detector, but such systems are notoriously imprecise, with many false positives. This would be intolerable, as it would either impede legitimate use or require too many people to analyze the reports in a timely fashion.

As with intrusion detection, there exists a conflict between security and usability. Specification-based intrusion detection [24] suffers from the same problems as access control due to its "binary" nature (either allow the action, or block it). Anomaly based intrusion detection [14] uses statistical variations whose thresholds can be altered to flag or ignore more suspected attacks, but the consequences are that either real attacks are missed, security administrators are quickly overwhelmed with false positives, or legitimate users are mistakenly denied access. Masquerade attacks [28] and insiders complicate this problem considerably.

An alternative is to find a means of protecting systems with a policy that defines both forbidden and allowed actions—as well as the actual countermeasures (*e.g.*, blocking and allowing)—as a spectrum of possible decisions rather than as a binary decision, similarly to how we define "insiderness" above. For example, rather than denying access to read a file, allow the access but log it. Rather than denying all access to a file system, restrict access only to reading, and then to only to reading a specific partition. We call this notion *optimistic access control* because the system is *optimistic*: it allows actions up until a particular security threshold is met. Beyond this threshold the effects of the actions are unrecoverable, so only then do the controls become *pessimistic* and block the actions. Optimism does not preclude other forms of protection. For example, an anomaly detection system might inform the triggers to various countermeasures.

Merging the ABGAC and the Unifying Policy Hierarchy constructions with optimism provides a framework for managing insiders using a spectrum of non-binary countermeasures. Determining how to apply the countermeasures is still a challenge. A natural countermeasure is forensic logging of events. Even in the physical world, financial transactions have long been allowed but recorded, and entry to physical facilities is often handled in a similar manner.

We have previously discussed the importance of computer forensics and the need for better solutions [33], in particular ones that use a systematic approach [34] rather than ad hoc solutions. *Laocoön* [37],[5] is a model of forensic logging and analysis that uses attack graphs based on intruder goals to impose a structure on log data, thereby describing the sequence of steps that take place and the data to show that these steps took place. The model can denote the set of events to be logged, prune unrelated data, aid in linking events into steps of an attack, and help bound the conditions that lead to an unusual or unexpected step in an attack. When implemented, the system can record forensic data at arbitrary levels of granularity, in standardized formats that are easy to parse. It can then correlate information, and prune away information not related to violations of the goals in question. If logging can be limited to the data represented in a model, then the analysis can be limited to only the data needed to understand the attacks of interest.

This approach leverages optimism to trigger forensic logging and thus reduce a human forensic analyst's labor while simultaneously making the system more usable. The result of merging Laocoön with optimism is the ability to apply different access control and countermeasures for different measures of risk and trust as defined in the ABGAC model. Thus, we have an improved means of conducting post mortem analyses of events. This does not mean that we can pre-classify all such events as good or bad—such is the providence of intrusion detection, not forensics—but we can use the logged data to determine what happened, and where in a chain of insider events, something may have happened which merits further investigation.

Recall that the Unified Policy Hierarchy defines ideal, feasible, practical, and run-time abstractions of a security policy. The gaps between these layers encapsulate limits on what each abstraction can specify. The ideal/feasible gap encapsulates technological limitations, the feasible/configured gap encapsulates efficiency and configuration errors, and the configured/run-time gap encapsulates implementation errors. The ideal/feasible gap is roughly equivalent to what Schneider referred to as *EM enforceability* [38] and also what is used by designers of high assurance systems to develop auditing subsystems [4, §24.3] based on precisely defined security policy models such as Bell-LaPadula [3] and Chinese Wall [10]. Additionally, the feasible/configured gap often relies in part on computational complexity; that is, what is efficient. These equivalences are important because EM enforceability defines the limits of what security policies can be enforced, and therefore defines the limits of which optimistic countermeasures (including logging) can be deployed. Also, while computational complexity may not define what is impossible (unlike EM enforceability), it can still guide a system administrator to define what is reasonable.

Merging the ABGAC and Unifying Policy Hierarchy constructions with optimism provides a framework for managing usable and flexible security policies on a diverse group of systems, with a diverse group of users (including insiders), using a spectrum of non-binary countermeasures. This approach leverages optimistic access control to allow policies on discrete computer systems in a way that they can

[5] Laocoön was the Trojan (an ancient detective of sorts) who recommended not letting the Trojan horse into Troy.

be merged together, and allow computation and data transmission to continue. Optimism itself can employ a variety of non-binary countermeasures, shifting between threshold values as indicated by the risk level from the ABGAC model.

A small example will demonstrate how all this fits together.

5.1 Example

In voting and elections, insiders, usability, and security all raise issues of security [36]. Consider how a voter casts a ballot in the United States. When distinguishing marks are made on paper ballots (*e.g.*, the voter signs the ballot), many jurisdictions do not count the ballot because stray marks may communicate the identity of the voter to an auditor. Laws in the United States prevent auditors from reverse-engineering the identity of a voter, because this enables both coercion of the voter and selling votes. On electronic voting machines, the same laws apply, but the problem, and consequently enforcement mechanisms, are more difficult to define. The goal of preventing the association of voters with ballots (as with paper ballots) conflicts with the need to record audit logs in detail sufficient to enable a forensic analysis to determine if something went wrong, and if so what caused the problem. But those logs may include information such as touches on a touch screen, the number of times that a voter has selected or deselected a particular candidate, or the number of times that a voter has visited a particular screen—ideal covert channels between the voter and the auditor. There are technological solutions to limit the capacity of this channel (such as adding noise or enforcing regularity to log data) [13, 15], but there are also solutions that consider procedural steps and the nature of insiders.

Consider how to apply Laocoön to provisional ballots, which highlights the problem of insiders. A provisional ballot is cast when the poll workers cannot determine whether the voter is entitled to vote. When paper is used, the voter marks their vote on a ballot normally. She then places the ballot in an envelope and seals it. This is given to a poll worker, who places the envelope in a larger envelope, writes the name of the voter and the reason for the provisional vote on the outer envelope, and drops it into the ballot box. When the votes are counted, provisional ballots are separated from the other ballots. One election official determines whether the ballot should be counted. If so, the election official removes the inner envelope from the outer envelope and passes it to another election official. That official removes the ballot from the envelope, and puts it with the other ballots from that precinct. Modeling this situation requires an Oracle Policy to dictate the allowable actions by the election officials. But, if implemented electronically, the method required for handling *provisional electronic ballots* relies on additional Feasible and Configured Policy constraints to devise a technological method for dividing the data. Thus there is a technological gap between the Oracle and Feasible policies, representing the challenge with enforcing procedural policies.

Now, working similarly with electronic ballots and audit logs, one problem is that auditors have access to all data. Thus, one possible solution is to divide the data in a way that separates the groups of people and the data such that no technical forensic analyst can see information that describes votes that were cast, and no non-auditor can view the log data. As above, this requires an Oracle Policy to dictate the allowable actions by both the auditors and the vote counters, and gaps exist with the Feasible and Configured Policies. Though this problem cannot be eliminated, it can be reduced by enforcing a policy that takes the threat into account. The Configured Policy is defined to start at the entry to the system, end at the data (the audit logs and ballots), and place bounds on the intermediate steps. The policy then monitors those paths to address the Oracle/Feasible gap. For example, the poll workers can be monitored with video cameras, and/or a "two person rule" requiring that no single person be left with the ballots at any time be enforced. All of these countermeasures—logging, monitoring, and the two-person rule—are also optimistic: they allow activity to proceed but with an effect to limit or monitor damage due to reduced trust.

As an alternative to modifying the system to detecting insider attacks as a means of enforcing a policy to address the gap between the Oracle and Feasible policies, the system can also be modified to detect specific vulnerabilities with regard to insiders [8]. For example, anything in the system that identifies the ballot uniquely, and associates it with the voter, can be eliminated. For forensic purposes, any unique item or number or code being given to the voter represents a potential vulnerability. Any time such a unique identifier is given to the voter (or by the voter), the fact that it is given should be recorded (*not* what it is, though—otherwise the forensic audit itself compromises secrecy and anonymity). Similarly, patterns in ballots can make the ballot uniquely identifiable, so ballots as a whole should be preserved. The dissemination of such unique identification can be used to trace its use later in the fault graph by looking through logs for evidence of communication of the unique identification.

Forensic evidence captured and used in the courtroom is another key where insiders and security are both important factors. For example, consider again our model of forensic logging. The model describes the data to be logged and the means to log it. But the model has its roots in technological events. What happens when the events are not attacker goals, but legal ones, such as preserving a chain of custody of evidence? Just as with voting, the technological and legal models must be merged so that any gaps between the two can be captured in the model and addressed through monitoring [35]. Additionally, the notions of protecting the integrity of the data and monitoring the paths to disrupt the integrity of must be addressed, as do the entry points into the "system" (*e.g.*, doors).

5.2 Summary

Taking technological steps to ameliorate the insider threat is challenging. Optimistic access control and rigorous applications of forensic logging provide several key benefits: a means for applying gradations of security enforcement, rather than simply binary enforcement; a means of applying security dynamically, rather than statically; and a means of providing more accurate results by gathering more information while delaying strong enforcement. The benefits are increased usability for legitimate users and more effective security against both inside and outside threats, without compromising either goal.

6 Conclusion

This chapter presents an alternate view of the insider problem. Rather than focusing on trust, the insider is defined by the ability to perform actions allowed by one policy level but disallowed at a higher policy level, or vice versa. This defines degrees of "insiderness," because different entities can be insiders in different ways. Alice may be an insider because she can read her manager's email. Bob may be an insider because he has physical access to the computer on which the email resides. In some sense, Bob's ability to access all the emails (not just those of Alice's manager) makes him more of an insider than Alice is.

This is the key point of this chapter. In some sense, the question of whether one is an insider is irrelevant. The critical characteristic is the degree of access that one has. That determines the threat, and moves us away from focusing on a (usually fairly arbitrary) definition of "insider." Instead, we examine who can access resources, and how, and what their psychological indicators are. The defensive techniques are the same for the insider and outsider who have the same degree of access.

Philosophically, this is satisfying because it simplifies the problem by eliminating a notion that is ill-defined, or defined in various ways, in the literature. Occam's razor is always satisfying to wield; whether the wielding of it in this case has simplified the analysis of the problem remains to be seen.

References

1. Accelerated learning through serious game technology (2008). URL `http://www.dodsbir.net/sitis/archives_display_topic.asp?Bookmark=34520`. SBIR OSD08-CR8: Human Systems
2. Band, S.R., Cappelli, D.M., Fischer, L.F., Moore, A.P., Shaw, E.D., Trzeciak, R.F.: Comparing Insider IT Sabotage and Espionage: A Model-Based Analysis. Tech. Rep. CMU/SEI-2006-TR-026, Carnegie Mellon University Software Engineering Institute (2006)

3. Bell, D.E., LaPadula, L.J.: Secure Computer System: Unified Exposition and Multics Interpretation. Tech. Rep. EST-TR-75-306, Electronic Systems Division, Air Force Systems Command, Hanscom AFB, Bedford, MA (1975)
4. Bishop, M.: Computer Security: Art and Science. Addison-Wesley Professional, Boston, MA (2003)
5. Bishop, M., Engle, S., Gates, C., Peisert, S., Whalen, S.: We Have Met the Enemy and He is Us. In: Proceedings of the 2008 New Security Paradigms Workshop (NSPW). Lake Tahoe, CA (2008)
6. Bishop, M., Engle, S., Gates, C., Peisert, S., Whalen, S.: Case Studies of an Insider Framework. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS), Cyber Security and Information Intelligence Research Minitrack. Waikoloa, HI (2009)
7. Bishop, M., Gates, C.: Defining the insider threat. In: Proceedings of the 4th Annual Workshop on Cyber Security and Information Intelligence Research (CSIIRW), pp. 1–3. ACM, New York, NY, USA (2008). DOI http://doi.acm.org/10.1145/1413140.1413158
8. Bishop, M., Peisert, S., Hoke, C., Graff, M., Jefferson, D.: E-Voting and Forensics: Prying Open the Black Box. In: Proceedings of the 2009 Electronic Voting Technology Workshop/Workshop on Trustworthy Computing (EVT/WOTE '09). Montreal, Canada (2009)
9. Brackney, R.P., Anderson, R.H.: Understanding the Insider Threat: Proceedings of a March 2004 Workshop. Tech. rep., RAND Corporation, Santa Monica, CA (2004)
10. Brewer, D.F., Nash, M.J.: The Chinese Wall Security Policy. In: Proceedings of the 1989 IEEE Symposium on Security and Privacy, pp. 206–214. Oakland, CA (1989)
11. Burdick, E., Wheeler, H.: Fail-Safe. Dell Puiblishing (1963)
12. Carlson, A.: The Unifying Policy Hierarchy Model. Master's thesis, University of California, Davis (2006)
13. Denning, D.E.: Secure Statistical Databases with Random Sample Queries. ACM Transactions on Database Systems **5**(3), 291–315 (1980)
14. Denning, D.E.: An Intrusion-Detection Model. IEEE Transactions on Software Engineering **SE-13**(2), 222–232 (1987)
15. Denning, D.E., Akl, S.G., Heckman, M., Lunt, T.F., Morgenstern, M., Neumann, P.G., Schell, R.R.: Views for multilevel database security. IEEE Transactions on Software Engineering **SE-13**(2), 129–140 (1987)
16. Director of Central Intelligence/Intelligence Community Staff Memorandum ICS 0858-90: Project SLAMMER Interim Report (U). Project Slammer is a CIA-sponsored study of Americans convicted of espionage against the United States. A declassified interim report is available at: `http://antipolygraph.org/documents/slammer-12-04-1990.shtml` and `http://antipolygraph.org/documents/slammer-12-04-1990.pdf` (1990)
17. Ferraiolo, D.F., Kuhn, D.R.: Role Based Access Control. In: Proceedings of the Fifteenth National Computer Security Conference, pp. 554–563 (1992)
18. Garfinkel, R., Gopal, R., Goes, P.: Privacy Protection of Binary Confidential Data Against Deterministic, Stochastic, and Insider Threat. Management Science **48**(6), 749–764 (2002)
19. Gelles, M.: Exploring the mind of the spy. In: Employees' guide to security responsibilities: Treason 101. Texas A&M University Research Foundation (2005)
20. Greitzer and Kangas. (personal communication)
21. Greitzer, F.L., Frincke, D.A., Zabriskie, M.M.: Information Assurance and Security Ethics in Complex Systems: Interdisciplinary Perspectives *(in review)*. In: M.J. Dark (ed.) Social/Ethical Issues in Predictive Insider Threat Monitoring. IGI Global, Hershey, Pennsylvania (2009)
22. Greitzer, F.L., Paulson, P., Kangas, L., Edgar, T., Zabriskie, M.M., Franklin, L., Frincke, D.A.: Predictive modelling for insider threat mitigation. Pacific Northwest National Laboratory, Richland, WA, Tech. Rep. PNNL Technical Report PNNL-60737 (2008)
23. Jones, A.K., Lipton, R.J.: The Enforcement of Security Policies for Computation. In: Proceedings of the Fifth Symposium on Operating System Principles (SOSP), pp. 197–206 (1975)

24. Ko, C., Ruschitzka, M., Levitt, K.: Execution Monitoring of Security-Critical Programs in Distributed Systems: a Specification-Based Approach. In: SP '97: Proceedings of the 1997 IEEE Symposium on Security and Privacy, p. 175. IEEE Computer Society, Washington, DC, USA (1997)
25. Krofcheck, J.L., Gelles, M.G.: Behavioral consultation in personnel security: Training and reference manual for personnel security professionals (2005)
26. Kubrick, S.: Dr. Strangelove or: How I learned to stop worrying and love the bomb. Distributed by Columbia Pictures (1964)
27. Lampson, B.W.: Protection. ACM Operating Systems Review **8**(1), 18–24 (1974)
28. Lunt, T.F., Jagannathan, R.: A Prototype Real-Time Intrusion-Detection Expert System (IDES). In: Proceedings of the 1988 IEEE Symposium on Security and Privacy, pp. 59–66. Oakland, CA (1988). DOI http://doi.ieeecomputersociety.org/10.1109/SECPRI.1988.8098
29. Moore, A.P., Cappelli, D.M., Trzeciak, R.F.: The "Big Picture" of Insider IT Sabotage Across US Critical Infrastructures (2008)
30. Ning, P., Sun, K.: How to Misuse AODV: A Case Study of Insider Attacks Against Mobile Ad-Hoc Routing Protocols. Ad Hoc Networks **3**(6), 795–819 (2005)
31. Parker, D.: Fighting computer crime: A new framework for protecting information. John Wiley & Sons, Inc. New York, NY, USA (1998)
32. Patzakis, J.: New Incident Response Best Practices: Patch and Proceed Is No Longer Acceptable Incident Response. Tech. rep., Guidance Software, Pasadena, CA (2003)
33. Peisert, S., Bishop, M., Karin, S., Marzullo, K.: Analysis of Computer Intrusions Using Sequences of Function Calls. IEEE Transactions on Dependable and Secure Computing (TDSC) **4**(2), 137–150 (2007)
34. Peisert, S., Bishop, M., Karin, S., Marzullo, K.: Toward Models for Forensic Analysis. In: Proceedings of the Second International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE), pp. 3–15. Seattle, WA (2007)
35. Peisert, S., Bishop, M., Marzullo, K.: Computer Forensics *In Forensis*. In: Proceedings of the Third International IEEE Workshop on Systematic Approaches to Digital Forensic Engineering (IEEE-SADFE), pp. 102–122. Oakland, CA (2008)
36. Peisert, S., Bishop, M., Yasinsac, A.: Vote Selling, Voter Anonymity, and Forensic Logging of Electronic Voting Machines. In: Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS), Digital Forensics – Pedagogy and Foundational Research Activity Minitrack. Waikoloa, HI (2009)
37. Peisert, S.P.: A Model of Forensic Analysis Using Goal-Oriented Logging. Ph.D. thesis, Department of Computer Science and Engineering, University of California, San Diego (2007)
38. Schneider, F.B.: Enforceable Security Policies. ACM Transactions on Information and System Security (TISSEC) **3**(1), 30–50 (2000)
39. Schultz, E.: A Framework for Understanding and Predicting Insider Attacks. Computers and Security **21**(6), 526–531 (2002)

Legally Sustainable Solutions for Privacy Issues in Collaborative Fraud Detection

Ulrich Flegel, Florian Kerschbaum, Philip Miseldine, Ganna Monakova, Richard Wacker, and Frank Leymann

1 Introduction

One company by itself cannot detect all instances of fraud or insider attacks. An example is the simple case of buyer fraud: a fraudulent buyer colludes with a supplier creating fake orders for supplies that are never delivered. They circumvent internal controls in place to prevent this kind of fraud, such as a goods receipt, *e.g.*, by ordering services instead of goods. Based on the evidence collected at one company, it is often extremely difficult to detect such fraud, but if companies collaborate and correlate their evidence, they could detect that the ordered services have never actually been provided.

There are many other cases with higher economical impact which cannot be detected by one party alone, *e.g.*, money laundering and insurance fraud. An aris-

Ulrich Flegel
SAP Research Center Karlsruhe, Vincenz-Prießnitz-Str. 1, 76131 Karlsruhe, Germany, e-mail: ulrich.flegel@sap.com

Florian Kerschbaum
SAP Research Center Karlsruhe, Vincenz-Prießnitz-Str. 1, 76131 Karlsruhe, Germany, e-mail: florian.kerschbaum@sap.com

Philip Miseldine
SAP Research Center Karlsruhe, Vincenz-Prießnitz-Str. 1, 76131 Karlsruhe, Germany, e-mail: philip.miseldine@sap.com

Ganna Monakova
Institute of Architecture of Application Systems, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany, e-mail: ganna.monakova@iaas.uni-stuttgart.de

Richard Wacker
Karlsruhe Institute of Technology (KIT), Institut für Informations- und Wirtschaftsrecht (IIWR), Haid-und-Neu-Strasse 6, 76131 Karlsruhe, Germany, e-mail: richard.wacker@kit.edu

Frank Leymann
Institute of Architecture of Application Systems, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany, e-mail: frank.leymann@iaas.uni-stuttgart.de

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_7,

ing challenge is therefore collaborative fraud detection where organizations employ fraud detection algorithms on their joint data. It seems that the prerequisite for collaborative fraud detection is data sharing, *e.g.*, as proposed in Section 2. Data sharing brings along with it a number of new security and privacy challenges. Even in a risk-neutral setting no party is inclined to share its data, if the benefit does not exceed the risks plus the costs involved. Precisely assessing the risks of data sharing is very difficult, in particular for fine-grained data, such as event logs, since it is unclear what can be inferred from the data. Furthermore decision makers are often risk averse, such that the perceived risks can be very large, and therefore many companies share data very reluctantly. If the data to be shared is related to people, privacy legislation often prohibits or at least regulates and limits the possibilities for data sharing. We examine the legal ramifications of detecting fraud in detail in Section 6. As a consequence collaborative fraud detection should be performed in a privacy-respecting way, minimizing the data sharing risks.

In Section 2 we give an introduction into monitoring modern distributed business systems built according to the *Service Oriented Architecture* (SOA) paradigm. Section 3 shows how evidence produced by monitoring such systems can be correlated by use of heuristic rules and proposes a suitable architecture for that purpose. In Section 4 we switch from the technical view to a legal perspective. We motivate analyzing the use of monitoring data according to six basic privacy rules in Section 5.1, forming a common denominator of pertinent privacy acts in use. These legally motivated rules are applied in Section 7 to technical systems for fraud detection, as described in Section 2 and 3. We comprehend the results from this analysis as the benchmark for improvement for technical privacy mechanisms for fraud detection. To be able to propose practical privacy mechanisms we investigate the main multilateral requirements for detecting fraud and derive technical requirements for privacy solutions for fraud detection in Section 7.1. In Section 7.2 and 7.3 we present appropriate mechanisms for pseudonymizing evidence for fraud detection. We switch again to the legal perspective in Section 8 to determine the improvement that the additional effort for the proposed privacy mechanisms can afford us, before we conclude in Section 9.

2 Monitoring Modern Distributed Systems

Service Oriented Architectures are a way for a software system to expose its functionality via components, called services. Each service encapsulates a set of functions, or actions that it can perform, which form its functionality. Standards have been developed to describe services from an operational perspective using WSDL [30] and abstract BPEL [22], from a semantic perspective describing the actions it performs using OWL-S [29] or WSMO [32], as well as from a non-functional perspective using WS-Policy [31]. To ensure correct operational behaviours were conducted, observations of a system are needed. This section analyses observational characteristics of a service, shows how these characteristics can be modelled by way

of the evidence that can attest to service execution, and discusses how the provided evidence enables detection of fraudulent service behaviour.

To specify the unified content the observational characteristics of every service should contain, different service types in a SOA are analysed by way of the actions they can perform, and their interactions. SOA systems can be layered based on the type of services provided as shown in Figure 1.

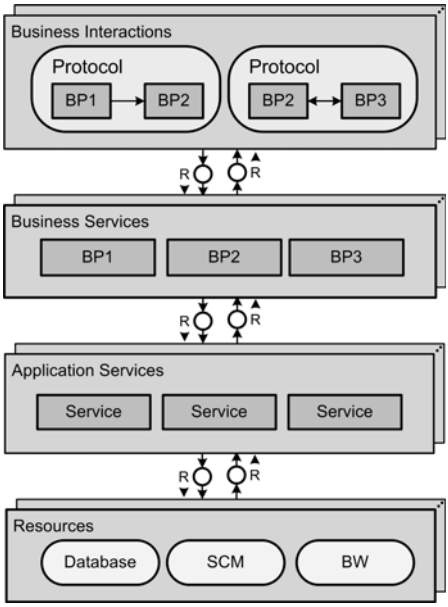

Fig. 1 Implementation levels of a SOA

The bottom layer contains system resources, which are accessed via *application services* shown on the next layer. Application services can be viewed as resource wrappers (or resource abstractions/virtualisations), which provide an interface enabling actions to be performed by the underlying resources.

Application services can be orchestrated to provide composite actions. Orchestrations define a flow of service invocations, commonly modelled as business process specifications. In classical business process design, as exemplified by the BPEL or BPMN standard [22, 25], services provide the functional implementation of business activities within a process, that lead to the achievement of a business goal. Business processes can be themselves exposed as services, encapsulating the composite action as a single action that can then be re-used within other orchestrations. As services perform their actions upon other services to achieve a business goal, they can be considered as *business services*. On this layer, the orchestrated services represent the application services upon which they perform, however do not directly expose these services. Instead, a business service can provide an action of type *in-*

voke that invokes the process represented, and thus perform the actions as defined in the process that are performed by the application services.

At all layers, services are invoked via the interface they expose. The service requester sends an invocation message as specified in a WSDL file. An Enterprise Service Bus (ESB) is a middleware component that acts essentially as a sink for these messages. ESBs can manipulate, redirect, and transform messages sent to and from services. This enables location transparency, which contributes to the flexibility of the SOA systems. Services can be dynamically discovered and chosen as long as the selection criteria, which include desired functional and non-functional properties, are satisfied. Viewing an ESB as a single logical component responsible for the message delivery, the complete service communication, also called service choreography [6], is visible in the ESB. Because an ESB itself can be exposed as a service, we specify the third type of services called *interaction* services, which are contained within the top layer in Figure 1. The interaction services perform, but are not limited to, actions *deliver* and *transform* on the messages which represent communication between business services.

Observations of service behaviours can be derived from multiple points within the SOA implementation, depending on the layer being analysed. For example, a service invocation that occurred within a business process can be observed via monitoring the business service encapsulating the business process (observing evidence signifying that the business process has invoked the service), via monitoring the ESB that captures the message sent to invoke the service (observing evidence signifying that the invocation message was delivered, accepted or deleted), and via monitoring the invoked service (observing evidence signifying the execution of the invoked operation).

As such, SOA provides rich sources of evidence attesting to system behaviour. This makes SOA systems particularly of interest to companies wishing to simplify compliance management, where observations of a system are needed to ensure correct operational behaviours were conducted. SOA systems however are not typically described by way of the evidence that can attest whether particular functionality was performed by a service. As stated previously, services are described by their functional and non-functional characteristics. The non-functional properties can describe security aspects of a service [21] or reliable messaging [23]. The observational characteristics however have been neglected. In the next section, we define an evidence model that describes such observational characteristics.

2.1 Evidence Model

The observational characteristics of a service describe its ability to provide evidence attesting its behaviour. The provided evidence, together with the evidence received from the other services in a system, describe the observational characteristics of a service.

A service behaviour is specified by actions a service performs, the order in which these actions occur, and conditions which enable or disable execution of certain actions. At runtime a service produces an instance of the specified behaviour. In a complex system consisting of multiple services, the overall behaviour of the system is based on the current behaviours of the participating services. Because the single services in a SOA are unaware of the other existing services (the loosely coupled principle of SOA), from the point of view of a single service the behaviours of the other services and effects produced by these behaviours form the operational context of the system. Because the execution of an action in a service can depend on the current system context, which in its turn depends on the behaviours of the other services, monitoring of the system as a whole is required to detect fraudulent behaviour of a service.

The current state of the system is defined by the current state of every single service in a system. The current state of a service is defined by the current states of the actions the service performs. To enable the monitoring of a complex system, the states of the participating services must therefore be observed. For this purpose a service must provide evidence on action state changes, which implies the state change of the service and the whole system. Because an action state change can cause a state change of resources the action operates on, the evidence must include information about resources the action operates on. To enable correlation and aggregation of the evidence from the different services in a heterogeneous environment, a common model of the evidence is required.

Existing service description standards do not provide a model rich enough to capture all information specified above. WSDL describes services in terms of operations they perform and messages they accept and produce. The order, in which the specified operations must be invoked, is not specified in WSDL. For the services which require stateful interaction and multiple messages, the order in which these messages need to be sent, called interaction protocol, must be specified in addition to WSDL. The interaction protocol essentially describes the ordering of operations in a service and WSDL specifies which messages are required for each operation. The ordering can be specified for example using abstract BPEL. The specification of the operations a service performs together with their ordering can be considered as description of the external behaviour of a service.

To be able to detect a fraudulent behaviour of a service however, the description of external behaviour alone is not sufficient. In addition, the internal behaviour of the service and provided evidence attesting to both external and internal behaviour must be specified. Currently there are no standard ways to describe the required information. One way to do this is to extend the existing descriptions of the external service behaviour with the specification of the internal actions performed on the invocation of a service operation. To enable the common understanding of the actions specified, they can refer to the shared vocabulary captured in an action ontology, which can be standardised for every specific domain. The evidence of the internal behaviour of the service can be provided in the form of events on the state changes of the actions, as it indicates the state change of the service, plus information the events can contain, including the description of the current state of the resources.

In general, a system in a SOA can be described as follows. Let Σ denote a SOA system, then $\Sigma = \mathscr{S} \times \mathscr{R} \times \mathscr{I}$, where $\mathscr{R} = \{R_1, ..., R_l\}$ denotes the set of all resources, $\mathscr{S} = \{S_1, ..., S_n\}$ denotes the set of all services, and $\mathscr{I} = \{I_1, ...I_m\}$ denotes the set of all interactions between services (or service choreographies).

Every service S_i performs a set of actions $\{A_{i_1}, ..., A_{i_k}\}$. At any point of time every action is located in a specific state denoting the action progress. Let Γ denote the set of all action states. The state function $\sigma_{\mathscr{A}} : \mathscr{A} \times \mathscr{T} \rightarrow \Gamma$ returns the state of an action at specific time point, where $\mathscr{T}$ denotes time. The state of the service S_i at time t is defined by the states of its actions: $\sigma_{\mathscr{S}} : \mathscr{S} \times \mathscr{T} \rightarrow 2^{\Gamma}$, or $\sigma_{\mathscr{S}}(S_i, t) = (\sigma_{\mathscr{A}}(A_{i_1}, t), ..., \sigma_{\mathscr{A}}(A_{i_k}, t))$

The state of the system is defined by the current state of the system services, resources and interactions. As the previous section shows, every resource can be abstracted through an application service, which captures all accesses to this resource. This means that any state change of this resource implies invocation of an action of the application service. Therefore any resource state change can be related to the state change of the corresponding action of the application service. Thus, the resource states of the system can be derived from monitoring application service action state changes. For example if an action *update* on resource *data* changes its state to *completed*, then it can be derived that resource *data* is in state *updated*.

All service interactions happen through the service bus. Thus, monitoring of the state changes of the service bus actions provides enough information to derive the current state of service interactions. A similar approach for monitoring predefined choreographies in the service bus was described in [17].

Thus, the complete state of a SOA system can be derived from the state changes of the actions on the application service level, business service level and the interaction service level. Therefore the evidence model presented in this section considers the evidence that can show state changes of the service actions and provide information about the current state of the resources the action operates on, which allows monitoring for the resource state changes.

Fig. 2 shows a graphical representation of the evidence model based on the action state changes.

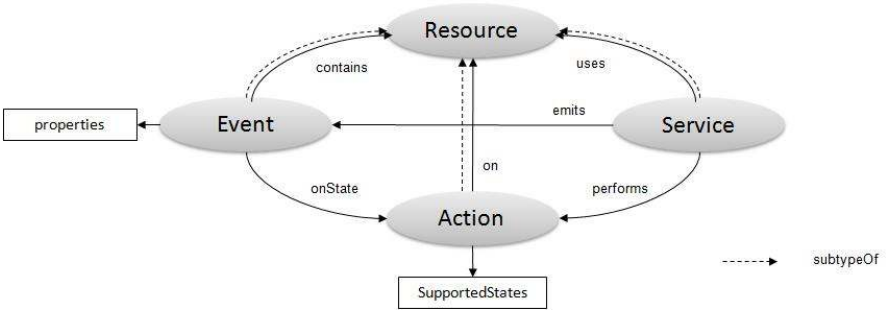

Fig. 2 Evidence model

The evidence model contains the following concepts:

- *Service* - represents the described service.
- *Action* - captures the actions service performs. A set of all actions builds action taxonomy. The sub- and super-class relationships between actions can be used to enhance modelling of the observational requirements. For example, if an evidence on execution of an action of type *AccessData* is required, and it is known that the actions *ReadData, UpdateData, DeleteData* are sub-classes of the action *AccessData*, then the evidence requirement can be propagated to all sub-class actions. Every action can specify a set of supported states as attributes. Every action can have its own set of supported states, we assume however that a superset of all possible states exists. This means that an action specific state set must always be a subset of the superset. An example of such a superset can be based on the BPEL activity state diagram [15] and consists of states *Started, Running, Faulted, Repaired, Suspended, Terminated, Completed* and *Compensated*
- The *Event* - describes events a service can emit which are related to a certain action state change. The relation *onState* between *Event* and *Action* concepts is an abstract relation. It can be refined with the *onStarted, onRunning, onFaulted, onRepaired, onSuspended, onTerminated, onCompleted* and *onCompensated* relations, depending on the states the corresponding action supports. Events can have properties, for example event timestamp. As a payload, events can contain information about resources the action operates on. This provides contextual information at the current execution state.
- *Resource* - Describes the resources which can be used by a service and on which the actions are performed. In an abstract way, everything can be considered as a resource: an action can be executed on a service, action or an event. Therefore a resource can be viewed as a super concept. Resources can have relations to other resources, which are captured in an ontology. A specific type of the resource ontology is an action taxonomy described above.

3 Observing Fraudulent Service Behaviours

A major advantage of SOA is its ability to connect multiple applications together to exchange and reuse functionality present in each. As an example, in the SAP Business Suite 7.0, many business processes are defined to provide Enterprise Resource Planning (ERP) behaviours over multiple SAP components. One such set of behaviours is defined for Accounts Receivable (AR) processes that record and manage accounting data of all customers for a business. This data forms the knowledge the business has about its customers, and is thus known as Customer Master Data (CMD). It is stored in a singular location, and is exposed via a set of services in the Business Suite.

Manipulation of Customer Master Data can yield multiple frauds. The SAP MIC (Management of Internal Controls) component for the Business Suite that controls AR processes defines a set of controls to prevent fraud. We consider how an example

of such a control can be implemented using the evidence model as a basis. The example control outlines how the action states introduced in the model can be used to implement data change management checks.

A control for AR data change management, which stops unauthorised systems changing sensitive data within the system, is detailed below.

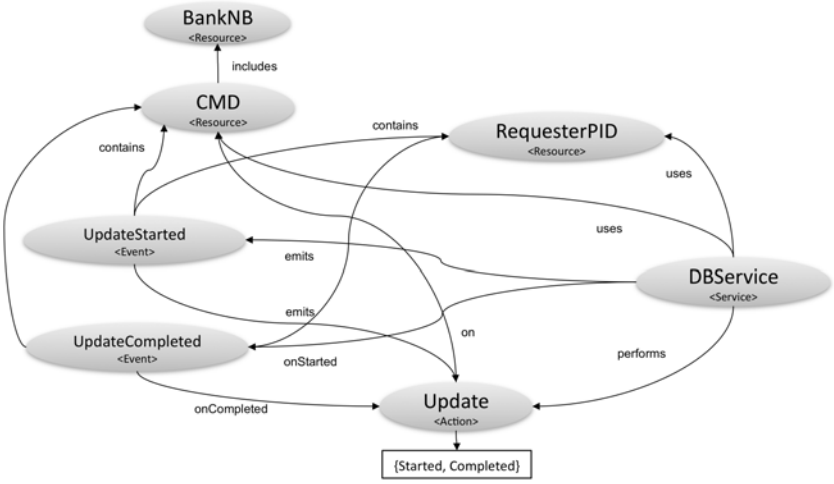

Fig. 3 Database Service evidence model instantiation

A/R 10.0 1 Access AR sensitive fields in ISP, CRM. *The creation and/or changes of the sensitive fields in the customer master data such as bank account references, payment terms are only possible in the R/3 System, Module FI/AR. Afterwards it is replicated into CRM. Creation or maintenance of customer bank references is only possible from the AR System.*

In this control, two main components, an SAP AR Service, and SAP CRM (Customer Relationship Management) are involved in a data exchange of CMD. We assume the data is stored in a database, CMDDatabase, which both systems share. A DBService is an application service that provides exposure of behaviours for this resource. There are two aspects to proving the control was correctly implemented. It should be captured when changes to CMD in the database are made. It should then be ascertained that only insensitive fields were changed, unless the update was made from the AR Service. It should be noted that the sensitive fields are defined

by the control, not the applications, and accordingly it cannot be assumed that SAP CRM is aware what is or what is not sensitive in terms of CMD.

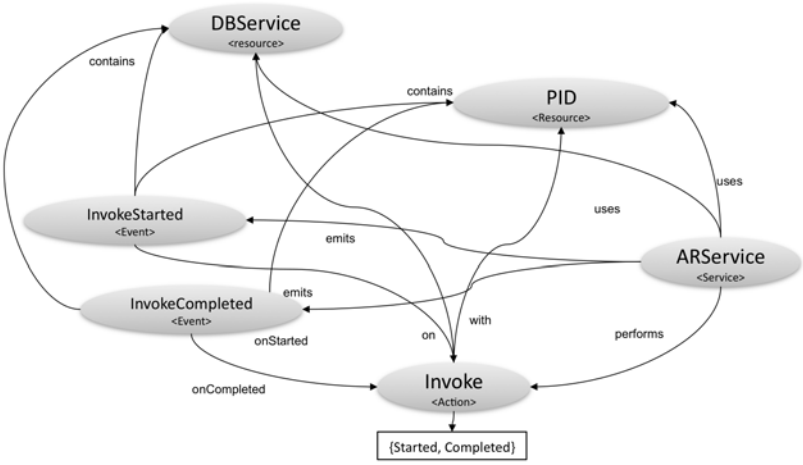

Fig. 4 AR Service evidence model instantiation

We assume events are produced by the DBService on the states start and complete on update action, with the state of the CMD record included into the event payload. For the control, we need to ensure that the update action for the CRM system does not add or change sensitive fields in the CMD record. We assume the sensitive field to be simply *bankNb*.

Using the evidence model, we define an abstract view of both components with regards to their use of CMD. To be able to monitor the specified control, we need to capture the events on start and complete of the update action in the DBService, including information of the sensitive fields at the current state and the ID of the service which triggered this update. Furthermore we need an event from the AR-Service indicating invocation of the DBService. The values of the sensitive fields in the events indicating the start and end of the update action must be equal, unless the update operation was invoked by the ARService. The graphical representation of the observational characteristics required to monitor this constraint is shown in Figure 3, where *PID* denotes the ID of the current service run and *RequesterPID* denotes the ID of the service run which invoked the DBService.

Assuming these events are provided, the monitoring rule can be specified as follows:

$$\forall e_1 \in UpdateStarted, \forall e_2 \in UpdateCompleted :$$
$$e_1.CMD.BankNb \neq e_2.CMD.BankNb \rightarrow$$
$$(\exists s_1 \in ARService, \exists e_3 \in InvokeStarted :$$
$$Emits(s_1, e_3) \wedge (e_3.PID = e_1.RequesterPID))$$

This example shows how correlation of the events from different services enables detection of a database update from a service different from the ARService. Note that the events can be adapted to operate in a synchronous manner, which would mean that the service which emits the event waits for permission to continue. In this case the security would improve, as some violations can be prevented by blocking the execution of a service action by observing a synchronous event on the start of an action, but service performance would suffer in this case.

3.1 Architectural Support

We now detail how such observation and evaluation could take place in practice. In Figure 5, a simplified architecture of a system is given that evaluates rules using evidence gathered from events described in the proposed model. The basic design of the system is described, with a description of the information flow.

The architectural schematic describes two entities: the Process Owner (in our examples, the *Hospital*), and a third-party in a Business Process Choreography (for example, an *Insurer*). As the process owner is unaware of the actual implementation of the services of the third-party, the architecture reflects that only a service model provided by the third-party is available for querying. We assume additional components are required for the service model to be managed, and exposed for query.

A business process execution component (here the *BPEL Engine*) instantiates a business process, and queries the *Rule Repository* to return predicates that govern the current execution state of the process. These predicates refer to the events described by the service models of each party in the choreography. Such predicates can serialised in an XML Query language with Window support to support temporal conditions. Such work is detailed in [5].

As the *BPEL Engine* is aware of which services are being invoked through the process, the *Repository* can build a list of rules that reference the events that can be produced by these services. It determines this relationship based on a query to each party service model manager. The matching rules are relayed to a *Rule Evaluator* who instructs a *Monitoring* component to inform it when events needed to evaluate the rules are observed. The *Rule Evaluator* instructs each service model manager to inform it when the events needed occur.

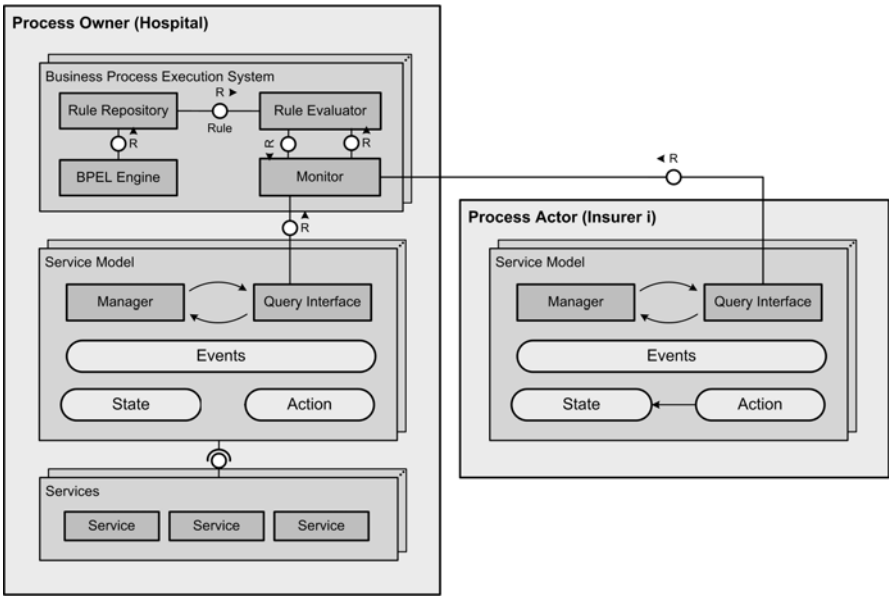

Fig. 5 Monitoring architecture

During the business process execution, services are invoked by the *BPEL engine* to provide the activities described in the process. Upon an invocation of an operation it provides, a service emits an event as described in its service model. This includes the properties that are represented in the event payload. The local service model manager informs the *Monitoring* component of this event if the *Monitoring* component indicated this event was of interest. This could be achieved in multiple ways: the use of a service bus for example, allows invocations and messages sent by and to services to be monitored. The *Monitor* forwards this to the *Rule Evaluator*. The event payload (detailed information regarding the event) is also sent so that the operational state leading to the event can be captured and represented in the rules too. The *Rule Evaluator*, upon receiving this event and the others it must observe to evaluate the rule, then evaluates the predicate.

4 Introduction to the Legal Perspective

Fraud often spans different organizations, and in the face of outsourcing, new fraud opportunities will be created. Detecting fraud requires a complete picture of the executed business processes, which can be implemented as described in Section 2. This necessitates collaboration of the involved organizations for detecting fraud. A main obstacle to collaborative fraud detection is data confidentiality or privacy where

parties are reluctant to share their possibly sensitive data. We analyse the statutory situation, and the effects of some fraud-detection specific privacy-enhancing technologies.

A fundamental problem in the discussion about international data privacy law is raised by the diverse viewpoints of data privacy as a result from different legal systems. A legal evaluation can hold only for a special legal system. On the other hand the objective, the protection of a citizen against the misuse or limitless propagation of his personal data is the same in any law system. Thereby aside from the dogmatic foundation the approach to constrain the spreading of personal data in a society leads to very similar basic ideas.

In consequence these basic fundamentals can be and have already been reduced by several institutions and jurisprudence to a set of basic principles, which can be seen as the least common denominator of data privacy law. Compliance to a set of rules is no guaranteed compliance to a special legal act, but it is surely an improvement towards a more privacy aware solution.

In many practical scenarios the processing of personal data is necessary and cannot be avoided completely. The processing of personal data serves the interests of the controller that in many cases are congruent to the interests of the person concerned himself – for example as indispensable element of an agreement. In consequence in most legal acts data privacy law allows handling of personal data in three cases. Firstly there is no need for a legal interference when the controller's interests impartially match the interests of the person concerned (see [8] Article 7, [12] §28). Secondly, if the controller's legitimate interests outweigh the interests of the person concerned. These cases arise, when one fundamental principle of democracy collides with another.[1] Any basic right of one person is limited by the basic rights of others. Thirdly the person concerned can legitimatize the handling of his data by an act of his own free will for a specified purpose.

5 Basic Principles of Data Privacy Law

In this section the approach to form generally accepted privacy principles shall be discussed, several examples shall be presented and finally it shall be tried to gain a condensed set of rules that cover the main issues of all presented examples.

Firstly it should be mentioned, that there are two different intentions that drive the development of simple rules instead of very complex and detailed legal regulation. On the one hand, the practical side often demands a less complex and universal set of rules to achieve a portable solution for, *e.g.*, software compliance. On the other hand the issue of harmonization in the field of data privacy is much more critical than in other fields of law. Information flow easily overcomes national borders an isolated application of data privacy acts is nearly ineffective. Data that is collected in a country with high data privacy standards can easily be transmitted into such coun-

[1] For example the data privacy right would undermine the right of expression.

tries, that do not regulate at all or where the national regulation is less restrictive. This is underlined by the titles of the most important harmonization attempts which often contain the *transborder flow* of data (see in [8] *free movement of data* and *transborder flows* of the OECD). To overcome these problems privacy acts usually contain norms that prohibit the transmission of personal data into countries without sufficient data privacy legislation, *i.e.*, the safe-harbour principle that is part of the EU-Directive 45/95/EC (see [8] Article 4 (a)). In times of global companies these rules are hard to monitor and it would be much more efficient to guarantee an international minimum standard of data privacy like the EU-directive in the European Union. As a result of different cultures, constitutions and viewpoints this minimum standard can hardly be achieved in the shape of a detailed legal act. Like in most multilateral political attempts it is more likely to settle on a set of basic ideas.

5.1 A Set of Six Basic Rules

We reviewed three guidelines to determine their essence in a form of common rules that agree with international privacy law: The *OECD guidelines on the Protection of Privacy and Transborder Flows of Personal Data* [24] reduce data privacy to a set of eight principles and inspired most of the other initiatives, such as the basic principles for Hippocratic Databases (HDB) [1], which are also based on the US privacy act of 1974 [27]. Bizer's *Seven Golden Rules of Data Privacy* [4] are mainly derived from the German data privacy law, the German conversion of the EU directive 46/95/EC [8].

As assumed, even though these rule sets originate or are inspired of different legal approaches, their basic ideas are widely identical. As basis for the evaluation of FDS, these ideas shall now be aggregated under six labels and briefly definied. They shall cover all aspects of data privacy that have been identified by now.

5.1.1 Data Avoidance

"The personal information that is collected should be exactly the amount that is necessary!"

The whole issue of data privacy arises only, if personal data is collected, stored or processed. The basic idea of balancing the interests of controller and person concerned claims, that there must be a legitimate interest of the controller for any single element of personal data that he stores. In other words, if there is no necessity for an element of personal data, it should not be collected or stored. The amount of data that is collected should be this minimum that really serves the controllers interests.

5.1.2 Transparency

"The controller should ex ante and ex post inform the person concerned about anything that affects the person's personal data."

It is beyond question, that the person concerned can only assert its rights or give his acceptance, when the controller is forced to inform ex ante about the kind of personal data, the purpose under which it is collected or stored and the recipients, that will or may get the data afterwards. In addition a controller has to inform ex post, when a person is interested in the personal information that is stored and in the authorization he has given. All these sanctions can be subsumed to a principle of transparency. A person concerned can only monitor the controller's actions, if these requirements are fulfilled. As a result of the directive of transparency it is a fundamental requirement that the controller has to store these facts in a way, that the person concerned is enabled to access them or at least get them from the controller, when he demands it.

5.1.3 Purpose Specification and Binding

"Any handling and the existence of personal data has to be strictly bound to a well documented purpose."

After the collection[2] or storage[3] of personal data this data moves out of sight from the person's point of view. The *Directive of Transparency* can only be adhered to, if the picture the controller draws of the handling of personal data is guaranteed to reflect what really happens with them. Therefore the controller has to be forced to ex ante specify the purpose for which the data shall be used and to strictly bind any handling of personal data to the specified purpose. In any case he has to justify in which way his handling of the data serves this purpose and why this handling could not be avoided. In addition the purpose binding has a temporal aspect. The permission to handle or store personal data is strongly tied to the purpose. If the reason for the storage of the data no longer exists, the data must be erased, unless there is no other justification.

5.1.4 Prohibition Without Explicit Permission

"Any handling of personal data that is permitted by legal permission or consent of the person has to be understood as an explicit exception of a general prohibition."

In the considered legal systems, permission for the handling of personal data can have its source either in the person concerned, who gives his consent to a specified handling of a specified quantum of personal data, or in law itself. The latter cases are

[2] Collection in the sense of data privacy law is acquiring data about a person concerned (see [12] §3 (3)).

[3] Storage in this sense is a process that results in personal data, without a visible action—*i.e.*, the application of a system log or installation of a video camera.

special circumstances where the interests of the controller and the person concerned do not differ[4] or where the legitimate interest of the controller in preventing harm to himself or his property impartially outweighs the interest of the person concerned.[5] Independent from the source (consent of the person concerned or legal permission) this authorization is an exception of a general rule that prohibits any handling of personal data. If the given circumstances do not exactly match the requirements of such an exception, the handling of personal data is prohibited.

5.1.5 Data Quality

"Any personal data that is stored and used should be kept accurate and up to date."

The controller has the obligation to keep stored personal data accurate and up to date. Data that is inaccurate or out of date must be corrected or deleted.

5.1.6 Data Security

"Personal data must be treated as sensitive data."

All these rules till now deal with the behavior of the controller, which is in any legal case someone, who is known and can be monitored by the person concerned. In cases, where the controller intends to transmit the data to someone else, adherence to the principle of transparency demands that he informs the person concerned also about the recipients. Aside from an intended action that can result in a misuse of personal information by the controller himself or a third person, there is still the possibility, that the data is stolen by a third party. To prevent that the controller has to treat personal data he stores as any other kind of sensitive data. State-of-the-art methods of data security need to ensure that personal information cannot be spied out.

6 General Legal Requirements of Fraud Detection Systems

In this section the general privacy-relevance of state-of-the-art FDS shall be legally evaluated. A special focus lies on the questions, if there is a legitimate foundation for the adoption of an FDS and what the source of this permission can be.

[4] *I.e.*, contractual relationships like ordering a product over the internet, where the data is needed for the delivery and similar cases.

[5] *I.e.*, a scenario where a person concerned as user damages the controller's property.

6.1 Privacy Relevance of Fraud Detection Systems

As a starting point of this analysis the question about the general applicability of data privacy law to Fraud Detection has to be answered more formally. The applicability of data privacy law is tied to the concept of *personal data*. If the event data of an FDS can be qualified as *personal data*, the data privacy law is applicable to these systems. The EU-Privacy-Directive 95/46/EG in article 2 defines personal data as "*... any information relating to an identified or identifiable natural person (´data subject´); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity.*"

A Fraud Detection System usually stores and processes information about the behavior of customers and insiders of an institution. While a customer may be a natural person, an employee definitely is a natural person. It is also beyond question that these persons are usually known to the controller – in general the institution that uses an FDS. The recorded data consists of elements that easily allow identification, so that any data record can be accounted to the person concerned.

The data that is collected by a fraud detection system is personal data by this definition and therefore underlies data privacy law. Thus the principles of data privacy law apply to FDS.

6.2 Necessary Data for Fraud Detection

A system that adheres to the principle of data avoidance needs to restrict the personal data that is collected to the minimum that is necessary to achieve the underlying aim. The data that is collected by an FDS can be limited in two dimensions. One dimension is the number of different data records. The other dimension is the number and the kind of data attributes that are stored. In both cases the limit of necessity cannot be fixed exactly. The only approach to this problem can be given by statistics. In the first dimension there might be a limit to an amount of data that is needed to detect harmful behavior with a specified probability. In the second dimension the limit is clearly exceeded, if data attributes are collected, that are obviously inappropriate to detect harmful behavior.

Summing up it can be stated, that a Fraud Detection System is no justification to observe arbitrary data that is accessible from the technical side. There must be a strong reference to the underlying purpose for any element of data that is stored. This could be achieved by identifying the actual data attributes evaluated by the FDS and only collecting these in cleartext.

6.3 Transparency in the Fraud Detection Context

The directive of transparency demands that the person concerned is previously informed about the kind and amount of data, which is stored by the controller. Even though the controller might consider it as inefficient to warn potential culprits, that they might be observed, the adoption of a Fraud Detection System should always be communicated to any person concerned. On the other hand detailed information of the methods that are used by the system would enable the other side to avoid detection. The extensiveness of details that have to be known to a person concerned therefore has to be found by balancing the interests[6].

In summary the adherence to the directive of transparency demands, that the controller informs about an adopted FDS system, but not about all technical details.

6.4 Purpose Specification and Binding in Fraud Detection

The purpose of the adoption of an FDS is by definition the detection of deceitful behavior. As one of the most important points an adopted FDS has to comply with the strict purpose binding principle. The personal data that is collected as basis for the search of behavioral patterns can potentially be used for a large number of other purposes, like for example monitoring work performance or analyzing ones personal habits and social relationships. Thus the use of collected data has to be strictly bound to the specified purpose. As another consequence data has to be erased or at least blocked directly after the fraud analysis is completed without raising an alarm involving that data.

To sum up the adoption of an FDS requires that the use of the collected personal is restricted to the purpose of detecting fraud.

6.5 Permissibility of Fraud Detection

The legal definition of fraud varies depending on the pertinent legislation. In general fraud can be defined as an "intentional deception made for personal gain or to damage another individual" (cf. German Criminal Code [13] §278 (1)). In the fraud detection scenarios this criminal behavior usually targets the controller. A manifested suspicion of criminal behavior is clearly a case, where the interests of the controller outweigh the interests of the person concerned. In the considered legal systems this suspicion would justify a processing of the data without consent of the person concerned. A problem arises from the fact that a fraud detection system,

[6] An FDS is thereby comparable with monitoring devices outside computer networks which possibly affect a person's privacy like video cameras. In a working environment in many countries there are other legal acts that require transparency about the existence and application of such systems. Transparency at least requires that the monitored person knows about being watched and recorded.

comparable to a video camera, stores data without such a manifest suspicion against the person in focus. On the other hand the controller is permitted to protect himself and his property under the condition of proportionality. This demands that it efficiently serves the purpose and there is no less inculpatory way of protection. Under this premise adoption of a FDS will be permitted on the basis of a legal permission. The prohibition as default principle demands, that if it is questionable, that a handling of personal data is allowed without the authorization of the person concerned it should be avoided. In other words, only if it is without question that a certain data attribute, processing step or the collection of additional personal information really contributes to a better detection of fraud, it may be done without consent of the person concerned.

In summary the legal permission for the adoption of FDS is bound to the legal concept of proportionality. The FDS must guarantee an effective way of protection and among the possible ways the least inculpatory.

6.6 Quality of Event Data

The data quality aspect in the underlying scenario is especially important. The data is stored to detect and prove criminal behavior. The impact of an inaccurate data basis can thereby possibly be a false suspicion against a person concerned.

6.7 Security of Event Data

Like in any other scenario, where personal data is collected, stored or processed the data that is collected or processed by an FDS has to be treated as sensitive data. As a result data safeguards must be adopted. With respect to the subject of fraud detection to detect and prove criminal behavior, the event data is of a high degree of sensitivity.

7 Technical Solutions for Privacy-respecting Fraud Detection

One might ask whether additional technical effort geared towards privacy can improve the overall legal situation with fraud detection and easen its application in a lawful way. We therefore switch back to a technical perspective and present mechanisms that we will analyse for their legal effect in Section 8.

Collaborative fraud detection, just as many other collaborative algorithms, can be performed in two main architectures in distributed systems. The first architecture (examples are [18, 19]) employs the help of a semi-trusted, central third party: every party collaborating in the fraud detection algorithm prepares its data locally, hiding

as much information as possible, *e.g.*, by pseudonymization or encryption, and sends it to the central party. The detection algorithm based on the input of all parties is then performed at the central party. The outcome, *i.e.*, the detected fraud cases, are returned to the parties. Different forms of result sharing can be imagined, *e.g.*, the third party learns the result or does not. Ideally the central party remains oblivious to the input data, *i.e.*, it is not trusted to treat data confidentially, but is trusted to perform the detection honestly. This central party may offer fraud detection as a service in the corresponding business model. In the second architecture (examples are [2, 33]) there is no third party and the parties directly interact. Powerful security techniques, such as Secure Multi-Party Computation [3, 14, 35], exist in order to provably protect the data of each party in this architecture.

These two architectures differ in a number of crucial aspects. By many the first architecture is considered to be more practical, since it requires significantly less communication and the necessary computations are simpler by an order of a magnitude. In recent years several implementations of collaborative detection algorithms have arisen [26], all following the first architecture. The second architecture for which algorithms with provable security exist, remain theoretical research, probably due to the expected high communication and computation effort. It is important to compare absolute numbers in this respect and not merely complexities in the "big-O" notation, since constants matter a lot.

We will provide approaches to solve central problems in collaborative fraud detection in the first architecture. For the second approach we analyse the security it provides. It can therefore serve as an example of the kind of trade-offs that need to be made in order to realize practical, privacy-preserving collaborative fraud detection. For the first approach we determine the improvement from a legal perspective in Section 8.

7.1 Technical Requirements

Considering overall legal and technical requirements, we identify the following four potentially conflicting goals:

1. *detection effectiveness* of cooperation alliances,
2. *privacy* of honest individuals,
3. further organizational *confidentiality* requirements of process actors and process owners, and
4. *efficiency*.

The solution for the general problem requires domain knowledge about the cooperation method to account for efficiency and cooperation effectiveness. For considering privacy and confidentiality, organization-specific knowledge is required. Particularly, in practice we need solutions with realistic and implementable trust requirements, because cooperation partners may be within different organizations.

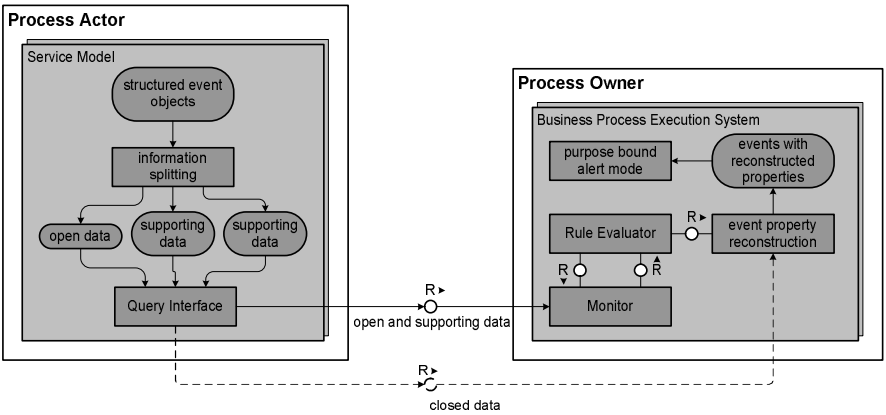

Fig. 6 Functional model for information reductions

Our solution extends the functional model from Section 3.1 (see Figure 5). A process actor service model generates *events* or just *information*, being consumed by its query interface (cf. Figure 5). The *process actor* represents the events or information using *structured event objects*. Appropriate information reductions process the structured event objects to satisfy detection effectiveness, privacy, confidentiality and efficiency. We distinguish lossless reductions and lossy reductions, depending on whether the original information from the structured event objects can be reconstructed or not, respectively:

Lossy reductions: remove information from structured event objects before forwarding it as *open data* to the process owner. The removed information must not be needed for further remote processing, and it should be definitely kept secret from the process owner and the other process actors, even under *inferences*. When information is coarsened, the detection effectiveness of the *Rule Evaluator* should not be affected unreasonably (see Figure 6, cf. Figure 5).

Lossless reductions: work by *splitting the information* contained in structured event objects into *open data* and *covered (masked, blinded) data* before forwarding it to the process owner (see Figure 6).

The open data of a lossless reduction is sufficient for the normal rule evaluation of the business process owner, possibly in conjunction with some *supporting data* that must be additionally generated depending on the evaluated rules. If a specific detection purpose is met during rule evaluation, a *purpose alert* is triggered. The respective open data together with the covered data allows for the reconstruction of the original information, subject to the detection purpose, *e.g.*, sufficient suspicion. The *data with the reconstructed information* can be used in the *alert mode*, *e.g.*, to hold fraudsters accountable.

7.1.1 Requirements for Open Data

The following requirements are crucial for sustaining the functionality of the relevant operations of the *Rule Evaluator* (see Figure 6):

R1: certain event properties (except for timestamps) need to be compared to certain event properties for equal content, or equal prefix content
R2: certain event properties (except for timestamps) need to be compared to values outside of the open data, *e.g.*, constant values, entries of a database
R3: distances of event timestamp properties need to be computed and compared to values outside of the open data, *i.e.*, a constant value
R4: the order of event timestamp properties needs to be determined

As a result, in order to sustain the effectiveness of the *Rule Evaluator* of the process owner, lossy reductions must be designed, such that they do not remove (timestamp) properties that are evaluated by the aforementioned operations.

Lossless reductions must be designed, such that the above operations can still be computed on the described event timestamps and event properties, and such that the results of the operations are still meaningful, *i.e.*, for an operation $\circ \in \{=_p, <, >\}$, where $=_p$ compares the properties up to a suitably determined prefix p, including $=_\infty$ for the full length, and two operands op_1 and op_2 in the open data and for a lossless reduction $r()$ holds $op_1 \circ op_2 = r(op_1) \circ r(op_2)$.

Note that sustaining the ability to compare event properties or operation results on event properties to values outside of the open data may require to provide *supporting data*, *i.e.*, the reduction $r_s()$ uses some parameter s, such that $op_1 \circ op_2 = r_s(op_1) \circ r_s(op_2)$ holds, where $r_s(op_1)$ is computed by the process actor and $r_s(op_2)$ is computed by the process owner.

7.1.2 Specific Requirements for Pseudonyms in Open Data

In the following, we consider pseudonymization as a special case of lossless reductions, and we focus on the specific requirements for pseudonyms in the open data, such that the *Rule Evaluator* of the process owner(s) sustains its detection effectiveness. Hence, the lossless reduction $r_s()$ replaces some property f with an appropriate pseudonym $r_s(f)$, where s is a parameter that can be used to generate distinct pseudonyms for f. Note that $r_s(f)$ needs to preserve the comparability of property prefixes, if required by R1, *e.g.*, [34]. Also note that pseudonyms traditionally are used to hide personal data, such as identifiers of users, but in a general scope we consider pseudonyms as place-holders for arbitrary properties. A pseudonym $r_s(f)$ is appropriate, if

- $r_s(f)$ respects the syntax constraints of the *Rule Evaluator* wrt. f
- $f =_p f' \Rightarrow r_s(f) =_p r_s(f')$ holds if R1 requires that f must be testable for equal content or prefix to an event property f'; note that both $r_s(f)$ and $r_s(f')$ are computed by the process actor

- $f \neq_p f', s \neq s' \Rightarrow r_s(f) \neq_p r_s(f'), r_s(f) \neq_p r_{s'}(f')$ holds generally, *i.e.*, $r_s()$ is collision-resistant, such that no unrelated events are correlated by accident
- $f = c \Rightarrow r_s(f) = r_s(c)$ holds if R2 requires $r_s(f)$ to be testable for equal content of a clear-text value c; note that $r_s(f)$ is computed by the process actor, who also provides s in the supporting data, such that the process owner can compute $r_s(c)$
- $f \neq c \Rightarrow r_s(f) \neq r_s(c)$ holds generally, *i.e.*, $r_s()$ is collision-resistant (see above)

Note that R2 can be required independently from R1 wrt. to f, such that R2 may be required in addition to R1. The process actor only needs to provide s in the supporting data, if R2 holds. Also note that a database lookup for a given $r_s(f)$ requires the process owner to compute $r_s(c)$ for all c visited in the database, until a match is found.

In order to reduce the inferences an attacker can make on the transitive closure of a given pseudonym, it is desirable to use $r_s()$ in a way, such that additionally

- $f =_p f' \Rightarrow r_s(f) \neq_p r_{s'}(f'), s \neq s'$ holds if R1 does not require that f must be testable for equal content or prefix to an event property f'; note that the process owner then does not need to and therefore is incapacitated to decide whether $f =_p f'$ or $f \neq_p f'$

We assume that all process actors a priori know the fraud detection rules of the process owner, such that all agents know, when R1 and/or R2 are required. This assumption can be met by proper coordination of the configuration of all parties.

All process actors that pseudonymize a given f must choose s in a coordinated way, if R1 is required. Then, the process owners can correlate events originating from distinct process actors by means of $r_s(f)$. If there is no coordination wrt. s, the following error can occur: $r_s(f) \neq_p r_{s'}(f'), s \neq s'$, despite $f =_p f'$, resulting in failure to correlate related events.

We have proposed concrete lossless reductions based on pseudonymization, which are respecting the requirements R1 and R2 [9]. The aspects for the corresponding covered data are summarized in Section 7.1.3 and 7.2. We have also proposed lossy reductions for event timestamp properties, which are respecting requirements R3 and R4. This recent result is described in some more detail in Section 7.3.

7.1.3 Specific Requirements for Covered Data

In contrast to the open data the covered data is basically independent from the rules evaluated by the process owner(s). This results from the fact that the covered data is not used by the *Rule Evaluator*. The covered data is used for information reconstruction when a purpose alert is triggered by the *Rule Evaluator*. This obviously results in fewer specific constraints for the design of lossless reductions.

It is necessary that the process actors generate covered data in a coordinated way. Note the analogy to the situation wrt. to the parameter s in Section 7.1.2.

7.2 Lossless Information Reduction with Covered Data

The process owner normally employs a *Rule Evaluator* to merely evaluate events with pseudonymized properties with respect to fraud rules. Only if a (*threshold*) *alert* occurs, *i.e.*, a fraud suspicion has been detected, the covered data can be used for *information reconstruction*, *i.e.*, the original event data can be reconstructed. In the *purpose bound alert mode* the process owner can employ the *reconstructed event data* to establish accountability for legal purposes, such as damage prevention and litigation. In our previous work on pseudonymization the pseudonyms that replace identifying properties in the event data are chosen randomly while respecting the requirements for linkability for fraud detection[10].

For the covered data, the approach leverages Shamir's threshold scheme for cryptographic secret sharing [28]: The fraud suspicions, also called *red flags*, for fraud detection are modeled as thresholds of secret sharing schemes. The covered data contains the encrypted identifying event properties that are replaced by the pseudonyms in the open data, and it contains shares of the respective decryption keys. As a result, the disclosure of the encrypted identifying event properties is enforced cryptographically, such that decryption is possible if and only if the pseudonyms are involved in a sufficient suspicion of fraud (*technical purpose binding*), *i.e.*, the number of shares associated with the pseudonyms exceeds the threshold in the model of the fraud suspicion. Note that it may be necessary to provide the ability to recover the decryption keys independently of a priori defined models of fraud suspicion in order to investigate fraud scenarios that have not (yet) been modeled. In that case, the grounds for decryption must be scrutinized by one or more trusted parties (*organizational purpose binding*). Involving these parties can be enforced cryptographically, *e.g.*, using threshold cryptosystems [11, 7].

7.3 Lossy Information Reductions for Timestamps

An important step of collaborative fraud detection is to identify correlated events. Time is often a necessary indicator for correlation, *i.e.*, one event happened (shortly) before the other. When collected as proposed in Section 2, events usually have a time property, *i.e.*, a timestamp. In order to be able to compare two timestamps from different sources in a distributed system, these systems require synchronized clocks. The wide-scale adoption of the Network Time Protocol (NTP) [20] offers such synchronization for systems connected to the Internet.

Our example protocol pseudonymizes timestamps for use of a central party in the first architecture (see Section 7). The central party – let's call her Trudy – should learn as little information as possible about the timestamps, but it should still be able to perform the functionality of timestamp correlation. Assume two events which have occurred at times t and t', respectively. Trudy's task is to compute

$$|t - t'| < \delta$$

where δ is a threshold not to be exceeded.

Let Alice and Bob be two process actors and Trudy be the process owner or a third actor conducting fraud detection. Trudy can compare their timestamps for them using the algorithm described in the following section. The security guarantees of the algorithm are

1. Trudy cannot learn the value of any timestamp, hence the information reduction is lossy.
2. Trudy can compute the distance between any two timestamps, if that distance is below or equal to the threshold δ.
3. Trudy cannot compute the distance between two timestamps, if that distance is above or equal to 2δ.

The notion of distance can be extended to the 2-dimensional case [16] which has applications in location-based services.

7.3.1 Architecture and Algorithm

Alice and Bob jointly choose a shared secret s which ensures that the cryptographic functions are sufficiently hard to guess for Trudy. Let $MAC(\cdot, s)$ denote a message authentication code with the key s. Alice and Bob also agree on the threshold value δ which is the maximum of distance comparisons. Note that δ limits the information Trudy can learn, but also limits the computations Trudy can perform. Furthermore, Alice and Bob jointly choose a random number r between $0 \leq r < \delta$.

For each of their timestamps t Alice and Bob perform the following steps:

1. Compute the lower grid point $l = \delta \cdot \lfloor \frac{t-r}{\delta} \rfloor + r$.
2. Compute the upper grid point $u = \delta \cdot \lceil \frac{t-r+1}{\delta} \rceil + r$.
3. Compute the distance m between t and l as $m = t - l$.
4. Compute the distance v between t and u as $v = t - u$.
5. Send the timestamp tuple $\mathbf{t} = \langle MAC(l,s), m, MAC(u,s), v \rangle$ to Trudy.

For simplicity we do not differentiate between l and u and their hashed counterparts and refer to both as grid points.

Trudy computes the distance $d = |t - t'|$ of two timestamps t and t' from the timestamp (tuples) $\mathbf{t} = \langle g_1, h_1, g_2, h_2 \rangle$ and $\mathbf{t'} = \langle g'_1, h'_1, g'_2, h'_2 \rangle$ as follows:

Case 1: $g_i \neq g'_j \forall i, j$: $d > \delta$

Case 2: $\exists g_c = g_i = g'_j$: $d = |h_i - h'_j|$

Imagine the timestamps on a ray from left to right. The grid points divide the ray into equal-sized sections. Alice and Bob compute for each timestamp the two grid points closest to the timestamp: l is the lower one and u is the upper one. In addition the distance from the grid points to the timestamp is sent to Trudy in plain-text, *i.e.*, the least significant bits are leaked, but the exact value of those bits is still blinded by r. Fig. 7 shows two timestamps t_1 and t_2 (as dots on the ray) with distance $d < \delta$ which have a common grid point g_c (grid points are shown as line markers on the

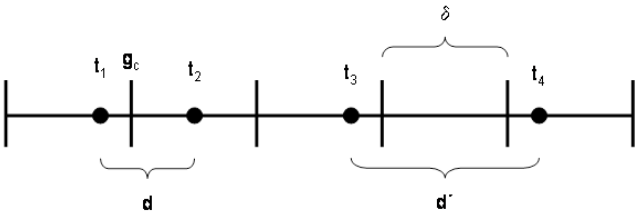

Fig. 7 Distances of 4 timestamps

ray) and two timestamps t_3 and t_4 with distance $d' > \delta$ which have no common grid point.

7.3.2 Limitations

A problem of our timestamp pseudonymization algorithm is that the privacy it provides is limited and only holds in the case of two timestamps. We describe an attack on larger sets of timestamps where multiple timestamps are close and form clusters that allow sorting all pseudonyms within a cluster.

Assume Trudy has a black-box device telling her for any two timestamp tuples their distance d, if $d \leq \delta$ or indicates otherwise. Such a device is a stricter form of our algorithm, which actually allows to compute the difference (and not just the distance) and also may allow the computation of some differences $d > \delta$ (but $d < 2\delta$). In summary, everything an attacker can do with such a black-box device, Trudy can do in our pseudonymization algorithm. Given this device and a data set T of tuples $t_1, \ldots, t_n$, the attacker can sort the timestamps aligning them on a linear ray. He selects two tuples $\mathbf{t}$ and $\mathbf{t'}$ from T by repeatedly querying the black-box device, until $|t - t'| = d \leq \delta$. He then searches the remaining set of timestamps for an instance t'' (again by repeatedly querying the black-box device), such that $|t - t''| = d' \leq \delta$. Now, he queries the device for $d'' = |t' - t''|$. We assume that $t < t'$. The attacker will learn this from the difference, but the distance hides the direction, yet it is one additional bit of information. We could have achieved the same in our difference computation by flipping just one coin and accordingly multiply each difference with 1 or -1 before sending it to Trudy. If $d'' \leq \delta$ which he will learn from the device, then if $d = d' - d''$, he concludes that $t < t' < t''$ or, if $d' = d - d''$, then he concludes that $t < t'' < t'$. If $t' - t'' > \delta$, he concludes that $t'' < t < t'$ and that $d'' = d + d'$, *i.e.*, he has computed the distance $'' > \delta$ by inference from two other distances $d < \delta$ and $d' < \delta$. Given enough data points the attacker sorts all timestamps along the time ray.

The problem becomes harsher in our algorithm, because we use grid points. The attacker only needs to align the grid points on the ray and the timestamps will follow, but our analysis shows that the sorting is unavoidable by any solution to the problem that abstracts the functionality the way we do.

7.3.3 Evaluation

The larger the distances we can compute, the larger the utility for fraud detection, but also the less privacy our algorithm provides. This subsection aims at estimating the limit on the privacy provided by our algorithm in terms of the parameter δ and the arrival rate of new events.

We can model the arrival of events (and corresponding log entries with timestamps) by a constant arrival rate λ, *e.g.*, $5\frac{1}{min}$. The distribution of the time difference between two consecutive events is then exponential with parameter λ and mean $\mu = \frac{1}{\lambda}$ ($12s$ in the example). The distribution remains exponential in the case of multiple collaborating sources, since the distribution of a random variable $Y = min(X_1, \ldots, X_N)$ is also exponentially distributed, if all X_i are exponentially distributed.

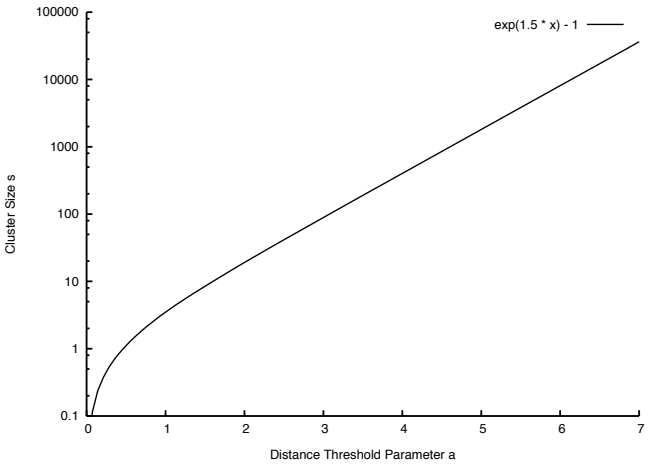

Fig. 8 Expected cluster size

We can express the threshold δ in terms of μ: $\delta = a\mu$ (*e.g.*, $\delta = 24s, a = 2$). Then using the cumulative density function of the exponential distribution we can compute the probability that an event occurs within time δ or less. We must model the interval $]\delta, 2\delta[$ where the probability that the distance $\delta + i$ $(0 < i < \delta)$ between two timestamps can be computed is $1 - \frac{i}{\delta}$ (proof omitted), so we consider the average of distances below or equal to $\frac{3}{2}\delta = \frac{3}{2}a\mu$ as computable. The probability p that a distance d of an event to its predecessor is computable (*i.e.*, $d \leq \frac{3}{2}\delta$) is then:

$$p = 1 - e^{-\frac{3}{2}a}$$

We call a consecutive series of timestamps $t_1, \ldots, t_n$ where the distance between two consecutive timestamps t_i and t_{i+1} is less than or equal to $\frac{3}{2}\delta$ a cluster. Due to

the attack described in the previous subsection the distances between all timestamps in a cluster can be computed by inference. A cluster is broken every time an event occurs with a distance $d > \frac{3}{2}$ to its predecessor and therefore the expected size s of a cluster is

$$s = (1-p)\sum_{i=0}^{\infty} ip^i = \frac{p}{1-p} = e^{\frac{3}{2}a} - 1$$

In our example ($a = 2$) the expected cluster size is $s \approx 19$ timestamps. Fig. 8 displays the expected cluster size for the threshold parameter a.

8 Legal Improvements by Pseudonymizing Event Data

We now revisit our analysis from Section 6 and determine the improvement afforded by the technical solutions proposed in Section 7. Generally the statements made in Section 6 about the legal requirements for state-of-the-art FDS still hold for the FDS operating on pseudonymized data, but as we will see, some of these requirements are now technically supported instead of relying on an organizational approach. The advantages of fraud detection systems that work on pseudonymized event data shall now be evaluated.

Therefore the relevant features of the method described in Section 7.2 will be summarized again from a legal point of view. Afterwards the impact on the adherence of identified basic aspects of data privacy law will be discussed.

8.1 Technical Description

The pseudonymizing technique introduced in Section 7.2 has the effect that the identifying elements of the personal information stored by the FDS are replaced by a randomly chosen pseudonym. The mappings between pseudonym and the original identifiers are encrypted with a key using a secure cryptographic technique before persisting them. Without this key the identifying data elements can only be decrypted by an attack on the cryptographic method which usually takes a disproportional amount of time.

The key is fragmented by a method that guarantees that the original key can be only efficiently computed, if all components of the key are known. (1) The key is fragmented into two components which are distributed to the data privacy official and the FDS administrator and (2) the key is fragmented and bound to the steps of a known suspicious behavioral pattern. Thus, the key can only be computed efficiently in two ways:

1. The FDS administrator and the data privacy official collaborate by putting their key pairs together (*organizational purpose-binding*).

2. All steps which belong to a known highly suspicious behavioral pattern have been executed by one user (*technical purpose binding*).

The aim is to ensure, that the pseudonyms can only by exposed, if there is a strong suspicion or evidence that the person concerned committed fraud. In the following section it shall be evaluated in which way such a technique can improve the compliance of an FDS respecting to some of the predefined principles of data privacy.

8.2 Privacy Relevance of Pseudonymized Event Data

At first, like in the more general evaluation of FDS, the question has to be answered, if preliminary pseudonymized event data is still personal data from the legal point of view. Many national regulations, for example the German BDSG define pseudonymized data per se as non personal data (cf. [12] §3 (6)). Pseudonymizing is defined as "*replacing identifying data elements by a label to prevent or significantly hinder the disclosure of the person concerned identity.*" It could be concluded that if the pseudonymized data would not meet the definition of personal data, the data privacy law would no longer be applicable. A problem arises from the legal scope of the controller who draws the line at the border of the company. As long as the pseudonymized event data, the user's identity and the mapping between both parts lie in the sphere of influence of one company the person concerned is still identifiable and therefore the data is still data privacy relevant. Thus the decomposed personal data is still legally qualified as personal data.

As just stated the reason why the pseudonymized data is still treated as personal data lies in the legal scope of the controller which per definition is the company. Based on this argumentation there is possibly an exception, if the technical and organizational architecture is created as follows:

1. Collection of event data and the fraud detection process are located in different organizations
2. The original event data is technically guaranteed to be never stored or dropped afterwards.
3. The key for the FDS administrator is automatically transferred to another company and guaranteed not to be saved locally.

In result the process actor that collects the event data – more precisely the data privacy official keeps just one part of the key for the organizational purpose binding. The other part lies in the hand of the process owner, or better, a third company that is conducting the fraud detection as a service, *i.e.*, executing the *Rule Evaluator* (cf. Figure 5). The pseudonymized event data is now from the perspective of both organizations non personal data per definition, because any attempt to indentify the user requires a successful attack on the cryptographic method that has been used.

In this special case the parts of the event data that can be disclosed with proportional effort is reduced to the portion that indeed form a manifest suspicion against the person concerned.

8.3 Strengthening the Data Privacy Official

In a normal FDS systems purpose binding can only be implemented by organizational provisions alone. In other words it depends on the privacy awareness of a number of people who potentially have access to the event data, to ensure a privacy compliant operation. The data privacy official in many national legal acts is defined as the person, who shall control the adherence to data privacy in an organization. As such he has at least to absolve training in this field of law (cf. i.e. [12] §4f). Practically he often has only a watchman's position, and is rarely involved actively into data privacy relevant processes like fraud detection, what leads to a situation, that these activities are often out of his attention.

By adopting the pseudonymizing extension of a FDS this role changes from a passive to an active one. The pseudonymizer shall ensure technically that the information can only be disclosed under certain conditions and by certain positions inside an organization. How effective this technically supported restriction is achieved depends on the trustworthiness of especially one person – the data privacy official itself, who takes a very important role in this system. The mode of his involvement depends on the way of disclosure, technical or organizational purpose binding. The organizational purpose binding restricts the disclosure of pseudonyms efficiently to cases in which FDS administrator and the data privacy official collaborate. So the data privacy official directly scrutinizes suspicion that needs be reached to disclose a certain pseudonym. In the technical purpose binding version the suspicion that allows disclosure depends on the technical definition of fraud that underlies the behavioral patterns that allow automated disclosure of a pseudonym. If the definition of the patterns the exposure of pseudonyms is strictly bound to the occurrence of an undesirable behavior of the "person concerned" this technical solution also guarantees an increase of data privacy awareness. Thus it must be ensured by organizational or technical means that the FDS administrator also collaborates with the data privacy official for defining the behavioral patterns.

In summary the adherence to the purpose binding principle can be significantly improved by a pseudonymizing extension because the decision about disclosure of personal data is concentrated in the position of the data privacy official.

8.4 Disclosure With Legal Permission

In the case of FDS in general the collecting, storing or processing of a personal data will rarely be legitimated by the given consent of a person concerned. Thereby the

disclosure is strictly bound to cases, where this technique meets the condition that the personal right of the person concerned is outweighed by the controller's interest in protecting himself and his property. For these cases the principles contain an explicit permission based on the underlying national law. Thus to be legitimate the data usage must be restricted to such purposes, where this exception is applicable. By concentrating the decision whether a certain pseudonym shall be disclosed or not, in the position of the data protection official, who is or should be an expert in this field of law, this principle is technically supported. Ideally and as a further extension for the organizational purpose binding the system should require a brief documentation about the facts that lead to his decision. In any case, if the system is adopted in an appropriate way the amount of personal data that is effectively revealed will be smaller than in a state of the art FDS.

In conclusion a system integrated guarantee for purpose binding strengthens the control over the collected data and the legitimation of its usage.

8.5 Data and System Security

While in a FDS without this extension the event data itself must be protected against misuse the privacy-respecting version leads to a situation where the pseudonymized event data is not critical any more. Thus the improvement to the purpose binding principle that can be achieved by pseudonymizing techniques depends on appropriate safeguards that hinder physical access to the machine and the pseudonymizer software. If the technique is not bypassed and the mapping information is inaccessible by a third person the original personal data can no longer be used for other purposes than fraud detection. In comparison to a state-of the-art FDS the adherence to data privacy principles demands, that personal data is held in a secure environment, but the safeguards that have to be implemented are mainly effective against outsiders. A protection against insiders of the controllers institution can only be guaranteed by organizational barriers. The pseudonymizing technique focuses on insiders. An outsider who tries to steal information usually tries to break into the account of an insider. Thereby, if this method minimizes the circle of internal users that have access to personal data, this effectively hinders outsiders too.

The presented pseudonymizing technique reduces the circle of persons inside the controller's institution that have access to the original event data. Hence, the technique also builds up an additional barrier for outsiders.

9 Conclusion

We have shown a general approach for monitoring modern SOA systems for fraud evidence and determined the legal ramifications. Fraud detection is subject to privacy law, implying obligations such as the limited use of collected evidence, infor-

mation of the affected individuals, and arguing the effectiveness and necessity of the planned FDS. For easening the introduction of FDS in practice we considered technical requirements for privacy mechanisms and proposed two concrete mechanisms for pseudonymizing event properties, one specialized for timestamps. We analysed the technical trade-offs of the latter and determined the improvement of the legal situation for the first.

The result is that the technical solution would significantly help the controller to guarantee, that a FDS adheres to the basic principles of data privacy. The strict purpose binding is technically guaranteed or at least supported. The level of data security has increased by implementing an additional barrier. From the legal point of view, especially from the perspective of the principle of proportionality, it can be concluded, that the availability of such a technique even constitutes an obligation to use it.

Acknowledgements

The research leading to these results has received funding from the German Federal Ministry of Economy and Technology under promotional reference 01MQ07012 (project THESEUS/TEXO) and the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement FP7-216917 (project MASTER). The authors take the responsibility for their contribution.

References

1. Agrawal, R., Kiernan, J., Srikant, R., Xu, Y.: Hippocratic databases. In: VLDB, pp. 143–154. Morgan Kaufmann (2002)
2. Atallah, M.J., Bykova, M., Li, J., Frikken, K.B., Topkara, M.: Private collaborative forecasting and benchmarking. In: V. Atluri, P.F. Syverson, S.D.C. di Vimercati (eds.) Proceedings of the ACM Workshop on Privacy in the Electronic Society, pp. 103–114. ACM (2004)
3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Proceedings of the 20th ACM Symposium on Theory of Computing, pp. 1–10. ACM (1988)
4. Bizer, J.: Sieben goldene Regeln des Datenschutzes. Datenschutz und Datensicherheit **31**(5), 350–356 (2007)
5. Botan, I., Kossmann, D., Fischer, P.M., Kraska, T., Florescu, D., Tamosevicius, R.: Extending XQuery with window functions. In: VLDB '07: Proceedings of the 33rd international conference on Very Large Data Bases, pp. 75–86. VLDB Endowment (2007)
6. Decker, G., Kopp, O., Barros, A.: An Introduction to Service Choreographies. Information Technology **50**(2), 122–127 (2008)
7. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: G. Brassard (ed.) Proceedings of the Conference on Advances in Cryptology (CRYPTO'89), no. 435 in Lecture Notes in Computer Science, pp. 307–315. Springer, Santa Barbara, California (1989)
8. Directive 95/46/EC of the European Parliament and of the Council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free move-

ment of such data. Official Journal L 281 (1995). http://europa.eu.int/eur-lex/en/lif/dat/1995/en_395L0046.html
9. Flegel, U.: Pseudonymizing Unix log files. In: G. Davida, Y. Frankel, O. Rees (eds.) Proceedings of the Infrastructure Security Conference (InfraSec2002), no. 2437 in Lecture Notes in Computer Science, pp. 162–179. Springer, Bristol, United Kingdom (2002)
10. Flegel, U.: Privacy-Respecting Intrusion Detection, *Advances in Information Security*, vol. 35. Springer, New York (2007)
11. Gemmel, P.S.: An introduction to threshold cryptography. Cryptobytes **2**(3), 7–12 (1997)
12. Federal data protection act. In: German Federal Law Gezette, p. 2954 ff. (1990). http://www.datenschutz-berlin.de/gesetze/bdsg/bdsgeng.htm
13. Criminal code. In: German Federal Law Gezette, p. 945 ff. (1998). http://www.iuscomp.org/gla/statutes/StGB.htm
14. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the 19th ACM Conference on Theory of Computing, pp. 218–229. ACM (1987)
15. Karastoyanova, D., Khalaf, R., Schroth, R., Paluszek, M., Leymann, F.: BPEL Event Model. Technical Report Computer Science 2006/10, University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Germany, University of Stuttgart, Institute of Architecture of Application Systems (2006)
16. Kerschbaum, F.: Distance-preserving pseudonymization for timestamps and spatial data. In: P. Ning, T. Yu (eds.) WPES, pp. 68–71. ACM (2007)
17. Kopp, O., van Lessen, T., Nitzsche, J.: The Need for a Choreography-aware Service Bus. In: YR-SOC 2008, pp. 28–34. Online (2008)
18. Lee, A.J., Tabriz, P., Borisov, N.: A privacy-preserving interdomain audit framework. In: Proceedings of the 5th ACM workshop on Privacy in electronic society, pp. 99–108. ACM, New York, NY, USA (2006). DOI http://doi.acm.org/10.1145/1179601.1179620
19. Lincoln, P., Porras, P., Shmatikov, V.: Privacy-preserving sharing and correlation of security alerts. In: Proceedings of the 13th USENIX Security Symposium, pp. 239–254. San Diego, California, USA (2004)
20. Mills, D.: Network time protocol (version 3) specification, implementation (1992)
21. OASIS: Web Services Security Policy Language (WS-SecurityPolicy) (2005). URL http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf
22. OASIS: Web Services Business Process Execution Language Version 2.0 (2007)
23. OASIS: Web Services Reliable Messaging Policy Assertion (WS-RM Policy) (2008). URL http://docs.oasis-open.org/ws-rx/wsrmp/200702
24. OECD: Guidelines on the protection of privacy and transborder flows of personal data. http://www.oecd.org/document/18/0,3343,en_2649_34255_1815186_1_1_1_1,00.html, (2009-07-01) (1980)
25. OMG: Business process modelling notation (BPMN) specification version 1.2 (2006)
26. Parekh, J.J., Wang, K., Stolfo, S.J.: Privacy-preserving payload-based correlation for accurate malicious traffic detection. In: Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense, pp. 99–106. ACM, New York, NY, USA (2006). DOI http://doi.acm.org/10.1145/1162666.1162667
27. United States House of Representatives 93d Congress, n.S.: US privacy act of 1974. http://www.usdoj.gov/opcl/1974privacyact-overview.htm (2009-07-01)
28. Shamir, A.: How to share a secret. Communications of the ACM **22**, 612–613 (1979)
29. W3C: OWL-S: Semantic Markup for Web Services (2004). URL http://www.w3.org/Submission/OWL-S/
30. W3C: Web Service Modeling Ontology (WSMO) (2005). URL http://www.w3.org/TR/wsdl20/
31. W3C: Web Services Policy 1.2 - Framework (WS-Policy) (2006). URL http://www.w3.org/Submission/WS-Policy/
32. W3C: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language (2007). URL http://www.w3.org/Submission/WSMO/

33. Waters, B.R., Balfanz, D., Durfee, G., Smetters, D.K.: Building an encrypted and searchable audit log. In: Proceedings of the 11th Annual Network and Distributed System Security Symposium (2004)
34. Xu, J., Fan, J., Ammar, M., Moon, S.B.: Prefix-preserving IP address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In: Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP), pp. 280–289 (2002)
35. Yao, A.C.C.: Protocols for secure computations (extended abstract). In: Proceedings of the annual IEEE Symposium on Foundations of Computer Science, pp. 160–164. IEEE (1982)

Towards an Access-Control Framework for Countering Insider Threats

Jason Crampton and Michael Huth

Abstract As insider threats pose very significant security risks to IT systems, we ask what policy-based approaches to access control can do for the detection, mitigation or countering of insider threats and insider attacks. Answering this question is difficult: little public data about insider-threat cases is available; there is not much consensus about what the insider problem actually is; and previous research in access control has by-and-large not dealt with this issue. We explore existing notions of insiderness in order to identify the relevant research issues. We then formulate a set of requirements for next-generation access-control systems, whose realization might form part of an overall strategy to address the insider problem.

1 Introduction

Most people have an intuitive, albeit informal, understanding of the terms "insiders" and "insider attacks". We all have seen spy movies in which agents and double agents exploit inside knowledge or privileged access to inflict damage on a hostile regime. The consequences of insider attacks can be extremely damaging. In January 2008, for example, Jerome Kerviel circumvented internal security mechanisms to place more than $70 billion in secret, unauthorized derivatives trades which, according to his employer Société Générale, resulted in a net loss of $7.2 billion to the bank [19].

It is less clear, though, how to formally define what insiders and insider attacks are, or what effective measures one could or should take to discover, prevent, miti-

Jason Crampton
Information Security Group, Royal Holloway, University of London, Egham, United Kingdom, e-mail: jason.crampton@rhul.ac.uk

Michael Huth
Department of Computing, Imperial College London, London, United Kingdom, e-mail: M.Huth@imperial.ac.uk

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_8,

gate or counter threats from insiders. For a small family-run business, for example, insider threats may pose a negligible risk. For a national intelligence agency, on the other hand, insider threats may be by far the biggest source of risk and potential damage. For these reasons, which we will discuss in more detail in Section 2, we argue that one should not seek to provide a single universally applicable definition of "insider" or "insiderness".

In this chapter, we mean to explore the insider problem *from the perspective of access control within IT systems*. We realize that this confines the insider problem to a single aspect of an organization's security mechanisms. Moore *et al.* recently produced a "big picture" report into insider sabotage [18], and made a number of interesting observations about the factors that might contribute to an insider attack. We recognize that some of these factors cannot be addressed or phrased in terms of access control: Observation 1 in [18], for example, states that "most insiders had personal predispositions that contributed to their risk of committing IT sabotage." We cannot reasonably expect access-control systems to detect or to be aware of such predispositions. Such issues are considered to be outside the scope of this chapter.

But some factors that contribute to insider attacks [18, Observations 5–7] certainly can be associated with access-control systems as we know them or as we can conceive them to be in the future. We begin by stating and discussing Observation 5:

> *"Observation 5: In many cases organizations failed to detect technical precursors."* where technical precursors are defined as *"an individual action, event, or condition that involves computer or electronic media and that precedes and is associated with malicious insider activity."*

Observation 5 is relevant for access control since many actions, events or conditions within IT systems are mediated through access-control mechanisms. For example, an insider might have had a strange work pattern, may have forgotten to do backup procedures, and may have moved vast amounts of data across unusual folder or account boundaries. The creation of audit trails that pertain to such actions, events or conditions is the standard practice for documenting activity that may be part of an attack or that may increase the risk of a future attack. In technical language, the access-control decision may have the side effect of creating an audit-log entry. But this side effect does not normally interact with the access-control system itself. Audit trails are mostly used for "post mortem" analysis, once an attack has happened. Indubitably, audit trails are valuable for such analyses.

But audit trails represent a genuine opportunity for making access control context-dependent, where the context is stateful and (partially) determined by the data contained in the evolving audit trail. In other words, any access-control system that makes authorization decisions dependent on technical precursors or previously recorded suspicious activity (as well as the usual authorization policies) will help in addressing the insider problem. Nevertheless, using audit data in this way gives rise to new challenges. In particular, the sheer volume of data in audit trails and the

lack of a *lingua franca* for the (efficient) creation, processing and interpretation of audit records represent considerable technical hurdles – in addition to legal hurdles or concerns.

Observation 6 notes that most insider attacks seem to happen when insiders are no longer "on the inside":

> *"Observation 6: Insiders created or used access paths unknown to management to set up their attack and conceal their identity or actions. The majority of insiders attacked after termination."* where an attack path is defined as *"a sequence of one or more access paths that lead to a critical system."*

Observation 6 is relevant to access control in IT systems since insider attacks based on IT systems necessarily need to go through access paths, be they legitimate or not. For example, two co-workers may decide to share their passwords for their user accounts, perhaps to circumvent some inconvenient system restrictions that "get in the way of getting work done." One of these co-workers may subsequently be fired and all his or her known access paths, such as personal user accounts, may be terminated with immediate effect. The disgruntled ex-employee may then launch a remote attack that crucially exploits his or her co-worker's user account in a hidden access path. A very similar scenario forms part of a hypothetical insider attack described by Moore *et al.* [18]. The access path was not known to management, which oversaw the termination of the attacker's known access paths, and the fact that a user shared a password with a co-worker to create this access path violated the organization's security policy. This raises at least two issues for access-control systems.

1. Such systems need to enforce an appropriate level of security whilst offering a sufficient degree of usability in the context of the particular organization's vital work practices. Otherwise people will invariably try to find ways that undermine the control of such systems in order to get work done. Establishing the correct balance between security and usability is increasingly recognized as an important aspect of effective security management [10].
2. The second issue is whether access-control systems could better detect or prevent hidden access paths. To illustrate, the sharing of passwords (an event that would occur completely outside of the scope of an access-control system) might be detectable indirectly.

To illustrate the second point, some office computer may be logged into a user account, doing routine office work whereas at the same time an office computer in another room is running a password-based authentication protocol for that same user account. These two contemporaneous events certainly indicate an abuse of this user account: the user identified with that account cannot be in two places at the same time. An access-control system, perhaps enhanced with an ability to process audit data, could detect this abuse and prevent the log-in attempt at the second computer.

One of the problems with such sophisticated access-control decisions is the possibility of (many) false positives. Even in the context of our example, the user may have to walk over to some other machine in order to launch a particular, secure application that can only be initiated from that machine, and he won't log out of the other machine as he will return to continue his routine office work. So one user uses two computers at the same time, but for legitimate reasons.

Observation 7 concerns the implementation of security within organizations:

> *"Observation 7: Lack of physical and electronic access controls facilitated IT sabotage"* where electronic access control is understood as *"the rules and mechanisms that control electronic access to information systems"* and physical access control is defined as *"the rules and mechanisms that control physical access to premises."*

Observation 7 is hardly surprising. In the context of insider attacks based on the misuse of IT systems it is evident that insider attacks are easier to plan, conduct, and conceal if no rules or mechanisms for restricting access to IT systems are implemented and enforced. More interestingly, perhaps, the above "lack of . . . access control" might also be interpreted as "lack of *effective* . . . access control". For example, a company may grant access to a sensitive database in its R & D department as follows: all employees have a standard, password-protected user account with user name as well as an identification number on their employee card; and the log-in for the database requires the user name and that id number, not the password for the user account. This gives insiders potentially wide access to all kinds of material in the R & D database: the user names of colleagues are pretty much public knowledge within an organization and it does not require great skills to look at an id card that is being swiped, say, at the organization's lunch cafeteria. Indeed, we had the opportunity to observe a very similar scenario in an actual organization.

Having introduced some of the issues of insider threats in the context of access control within IT systems, we now give an outline of the structure of this chapter.

Outline of chapter.

In Section 2 we motivate this work and discuss relevant related work. In Section 3 we introduce an alternative trust-based perspective on the insider problem. In Section 4 we identify requirements that arise from this new definition and sketch how we can realize these requirements by leveraging recent work in access-control policy languages. In Section 5 we discuss the sort of architecture that will be required to support access-control frameworks that are insider-aware. We conclude by discussing additional related work and our plans for future research in Section 6.

2 Motivation and related work

We have seen that there is evidence to suggest that the inadequacies of existing access-control mechanisms [18], whether automated or not, often play a crucial part in the successful launch of insider attacks. Apart from this empirical data, there are *technical reasons* to believe that a study of the insider problem from the perspective of access control in IT systems may be fruitful:

1. most of today's IT systems implement some form of access control – for example, to identify authorized users of the system or to limit the actions that each authorized user may perform – and understanding an organization's electronic access-control systems should help in discovering insiders, their degree of insiderness, and the level of risk they would represent;
2. an understanding of insiderness and insider threats in terms of access-control privileges and access history may allow IT systems to adapt access control in order to mitigate, counter or even prevent insider attacks;
3. the recent advances in *policy-based* access control may be leveraged to develop access-control frameworks in which policies can evolve dynamically or be retrofitted to deal with insider threats;
4. such policy-based access control systems may then be able to incorporate quantitative methods for the early detection of insider threats (such as host-based anomaly detection) in order to prevent, or limit the damage of, insider attacks.

It is also our hope that the understanding of the insider problem gained from studying the well-defined and structured context of policy-based access control may be transferrable to policies that are merely descriptive and not enforced within IT systems. Jerome Kerviel, for example, managed to launch his attack because, over time, he was able to take on two roles that should not have been held by any single individual (even at different points in time). But this role-based separation of duty [26] was not implemented in any part of the IT system. Support for role-based access control (RBAC) within that IT system, in combination with proper identity management, may have prevented or at least detected this insider attack.

We explore the potential of these technical ideas in later sections. Now we introduce three examples that further illustrate the diversity of the insider problem. We then discuss some existing work on defining insiders and the insider problem, concluding the section with a discussion of previous work on access control and the insider problem.

2.1 Illustrative scenarios

Inevitably, any organization of even moderate size and complexity must trust (some of) its employees with sensitive information and resources. For this reason, most organizations have security policies that specify who is authorized to access what resources. To ensure that these policies are enforced, all modern operating systems

and many applications provide authentication and authorization services. In using these services, an organization allows a number of individuals access to privileged knowledge of mission-critical information, and special access rights to mission-critical resources or both. To us, the *insider problem* refers to the inherent threats that organizations face when granting individuals such special privileges. Here are some examples of what many people would consider to be insider problems, the first one corresponding loosely to a case reported in [18]:

1. a system administrator may be given the right to manipulate folders that contain a substantial portion of her organization's intellectual property;
2. urgent building work at a company's headquarters may require contractors, employed by a third party, to be given permission to enter security-sensitive parts of that building; or
3. the personal assistant of a chief financial officer (CFO) would have access to the CFO's diary and contents or patterns of communication.

Part of the problem is that these individuals enjoy some particular trust relationship with the organization:

1. the system administrator *could* encrypt all files with a secret key in order to extort money from her employer, but *she is trusted not to do so*;
2. the building workers *could* be IT security specialists from competitors who want to infiltrate the company's premises and intranet, but *are assumed not to be*;
3. the personal assistant of a senior officer *could* divulge the existence and status of confidential merger talks, but *is expected not to*;

Organizations have to trust individuals, otherwise we cannot distinguish between authorized and unauthorized users. However, it may not be possible to articulate optimal trust relationships from a perspective of security for a number of reasons: lack of software support (*e.g.*, for the enforcement of role-based separation of duty in the attack by Jerome Kerviel), excessive cost, staff constraints, etc. Practical and economic considerations are often reason for compromising on security. Probst & Hunker [25], for example, note that many organizations take basic steps toward preventing insider attacks but rarely engage in addressing serious insider attacks; moreover, they argue that this appears to be rational economic behavior. In the context of the above scenarios:

1. A small start-up company may simply not be able to afford several system administrators, thus leaving the entire management of stored intellectual property (including backup procedures) to that sole administrator.
2. It may be impractical and expensive to vet every contractor that enters the building. It is much more cost-efficient to rely on (trust) the third party contractor to employ reliable staff.
3. A CFO naturally insists on a convenient single point of contact for arranging meetings, taking minutes, etc. But such convenience increases the risk and threat level of an insider attack by the personal assistant.

The above examples already indicate the difficulty of obtaining a working definition of "insider" that will fit all instances of the "insider problem". This has already been emphasized by Bishop in [3]. To illustrate this difficulty, it is too simplistic to say that every and only employees of an organization are insiders for that organization, considering that the above builders might be employees of an "outside" contractor but do have physical access to the "inside", the company's headquarters.

Even with a working definition of "insider" in hand, it is not clear how to put that definition into use so that it may aid with the detection, mitigation or countering of insider threats. Moreover, given the limited budget that is available for security solutions, there is economic pressure for research in this area to deliver solutions that can be deployed and managed in a manner that is perceived to be economically viable.

2.2 Definitions of insiders

Matt Bishop organized a panel at NSPW'05 on the insider problem and noted the diversity of existing definitions of insiders [6], which we cite here: Brackney and Anderson define an insider to be *"someone with access, privilege, or knowledge of information systems and services"* and a definition of the insider problem as *"malevolent (or possibly inadvertent) actions by an already trusted person with access to sensitive information and information systems"* [4]; in contrast, Patzakis defines an insider (implicitly) as *"anyone operating inside the security perimeter"* [20].

The summary of the recent Dagstuhl Seminar on countering insider threats [2] proposes several definitions of an insider, *e.g.*,

1. as someone "defined with respect to a resource, leading to *degrees of insiderness*[1]"
2. as "somebody with legitimate", past or present, "access to resources"
3. as a "wholly or partially trusted subject"
4. as "a system user who can misuse privileges"
5. as someone with "authorized access who might attempt unauthorized removal or sabotage of critical assets or who could aid outsiders in doing so".

It should be clear that the protagonists in the scenarios described earlier fit some of these definitions better than others. Any attempt at a comprehensive definition of insiders will have shortcomings. For example, if an "outsider" manages to break into an IT system by masquerading as a legitimate inside user, does this constitute an insider attack? In Section 3 we offer our own definition, which is based on and recalls definitions of trust and trustworthiness from the 1970s.

[1] Our italics.

2.3 Access control

For the sake of completeness, we now provide a brief description of what we understand by access control. In order to implement enterprise security policies, every attempted interaction between an authorized user and a protected resource (which could be as specific as a particular entry in a database table or as general as a computer system) is "trapped" by the access-control system. This interaction is modeled as an *access request*. The access-control system determines whether the access request is *authorized* (according to some policy and relevant security-configuration data) and then either allows the interaction to proceed (if the request is authorized) or prevents the interaction otherwise. Hence, an access-control system typically comprises:

1. a *policy-enforcement point* (PEP), which traps attempted interactions, generates access requests and enforces authorization decisions;
2. a *policy-decision point* (PDP), which accepts access requests and returns authorization decisions; and
3. a *policy repository* (PR), which stores authorization policies.

The typical architecture of an access control system is illustrated in Figure 1. The user and resource information is typically provided by trusted entities, perhaps the simplest example being (trusted) operating-system components such as an authentication service and a resource manager.

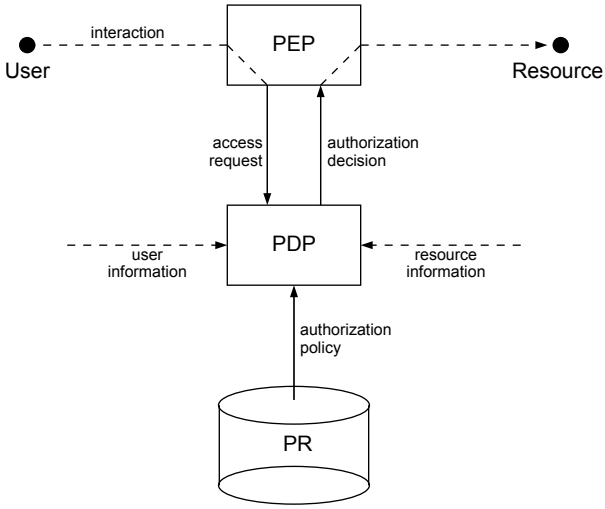

Fig. 1 A generic high-level access control architecture

In order to specify an access-control model, therefore, we need to define a language for articulating authorization policies and an algorithm that is used to evaluate access requests with respect to such policies.

2.4 The insider problem and access control

It can be seen that many problems that arise when discussing the insider problem are related to abuse of privileges granted by authentication or authorization services, both of which may be regarded as providing some form of access control. So how can an understanding of access control enable us to better understand and address the insider problem? There is very little existing work in this area, although Park and Giordano suggest the following issues need to be addressed in order to counter the insider threat in large-scale, dynamic IT systems [23]:

1. The management of privileges should not be based on identities, as this won't scale to distributed, highly dynamic and heterogeneous systems. The authors propose RBAC as a potential solution.
2. Access control needs to also support finer levels of granularity. For example, most systems grant access to an entire structured file and not, say, to specific fields in that file.
3. Most approaches to access control decide an access request by a static and predefined set of rules. Rather there is a need for *context-aware* access control, which makes decisions dependent on context.

We believe that these requirements are rather generic and do little to address the insider problem. Indeed, it is hard to believe that RBAC, which is by design unconcerned with individual users, is an appropriate underlying model for an access-control system that provides some protection against insider threats. Moreover, there is little evidence to suggest either that existing access-control systems are insufficiently fine-grained (see such systems for relational database management systems, for example) or that having more fine-grained access-control systems would help counter the insider threat in general. While we would agree that access-control systems do need to be context-aware, we believe that any such system would still need to be governed by some pre-defined (policy) rules.

We also note the existence of some preliminary work on combining risk and access control [8] and on using trust to assign users to roles [7, 16]. However, these efforts have a rather narrow focus, respectively on Multi-Level Security and Role-Based Access Control.

In the next two sections, we formulate an alternative definition for the insider problem that is more directly related to access control. We then discuss how recent advances in trust and reputation systems, and access-control policy languages can be used to address this revised insider problem.

3 Trust, trustworthiness, and the insider problem

Let us revisit the notion of "trust". It is generally accepted that trust is an assumption about the way in which an entity will behave (in a particular situation). In the context of the design of secure systems, we must assume that certain elements of our "system", such as the hardware, will behave in a particular way. These elements are called collectively the *trusted computing base* (TCB) [11].[2] It is imperative that the TCB operates in the way in which it is assumed or intended to operate (otherwise our trust is misplaced). One of the primary goals of the designers of secure operating systems is that the code that must be included in the TCB (because it is authorized to execute privileged instructions) can be formally verified [11]. Of course, it is rare for this goal to be realized in full because of the difficulty of formally verifying programs in general.

In the context of access control, there is an assumption that authorized users will behave in a certain way: that is, authorized users are assumed – that is to say, *trusted* – not to abuse the privileges for which they are authorized. The simplest example is that it is assumed (must be assumed, in fact) that a user will not divulge his username and password to another user. Clearly, however, some authorized users are not *trustworthy*. This suggests a more abstract and formal account of the insider problem:

> **TW:** The insider problem arises because the set of users *trusted* in some particular context is not equal to the set of *trustworthy* users for the same context.

This definition echoes the discussion of trust relationships and trust assumptions in the context of software security, made by Viega & McGraw in [27, Chapter 12]. Moreover, our scenarios and the proposed definitions of insiders given in Subsection 2.2 share a concern for resources and the interactions between users and those resources, where these interactions could modify, create or destroy resources.

Recognizing that insiders are somehow trusted with managing and transforming resources responsibly, it may be fruitful to focus on access requests and the potential threats that granting such requests may pose to the tactical or strategic aims implicit in the underlying IT system of an organization. For example, instead of asking whether a masquerading outsider is an insider or not, we may want to focus on the risk and probability that a specific inside user account has been compromised by that outsider, and let our access-control systems adapt appropriately should such a compromise be detected.

[2] Other electronic security "systems" also assume that certain entities are trusted. "Trusted third parties" are widely used in cryptographic protocols and public-key infrastructures, for example.

3.1 Insiderness

This emphasis on resources and on access rights to resources has led to the notion of *degrees of insiderness* [1], where the more privileges and skills a person has, the more risk he or she represents. However, we would argue that it is the abuse of an authorization that poses a risk and that this risk is some function of the authorized action, the trustworthiness of the user and relevant contextual information (which might include, for example, previously authorized access requests). There is no need for different levels of insiderness, just an ability to account for different levels of trustworthiness.

The degree of insiderness of a user has also been regarded as the threat level posed by the user [1]. Again, we do not regard this notion of insiderness as particularly helpful: it seems intuitively reasonable to regard a user that is known to represent a significant threat (for whatever reason) to be necessarily less trustworthy.

3.2 Trust management and risk assessment

We now briefly discuss connections between our formulation of the insider problem given in (**TW**), trust management systems, and risk-assessment frameworks. In particular, there is a natural correspondence between trustworthiness and the probability of misuse of authorizations. This suggests that trustworthiness and existing risk-assessment methodologies could be combined in a risk-aware access-control system. We will elaborate and illustrate this point in the next section. Moreover, trust-management and reputation-management systems could be regarded as tools to compute trustworthiness (not *trust*).

This connection is illustrated by similarities with problems and known attacks in reputation systems [12]. In a *whitewashing attack* against a reputation system, for example, the attacker exploits system vulnerabilities to repair the damage to his reputation that resulted from previous misbehavior. Similarly, a malicious insider may be able to manipulate audit logs in order to hide evidence of breaches of trust and to thus maintain his trustworthiness.

Another important aspect of risk and trust management in access control is that any relevant information about the possible detection of an insider attack needs to be reported to the access-control system so that it can adapt its behavior in an appropriate, context-aware fashion. For example, a person who received a formal reprimand for having shared the password for their user account with a co-worker would typically still have the same access-control rights and privileges as before. The damaged reputation is only reflected in their (paper) personnel records, not in the IT system through which they access sensitive electronic resources – and this mismatch may increase the risk of future insider attacks.

3.3 Pragmatics of identifying suspicious events

The key problem then is to identify (and then report) suspicious events. In the realm of social interaction, humans are quite good at identifying suspicious activities. Humans can differentiate well between anomalies that have context-aware explanations, and anomalies that can probably not be "explained away".

Machines, on the other hand, are good at recording events but less good at interpreting their significance – especially in dynamic contexts. This has long been recognized in the Artificial Intelligence community, *e.g.*, in the so-called *Frame Problem*. The movement or destruction of large amounts of data, *e.g.*, may just represent a yearly "housekeeping" exercise or it may indeed represent the prelude to a major attack. How would an automated policy or monitoring program know?

The challenge here is to develop automated systems that detect, record and collate all events that could indicate suspicious activity from authorized users. However, such systems must not generate too many false alarms, as such alarms will either lead to denial of access and inconvenience to well-behaved users, or to human intervention and analysis, which is costly and is not viable in large-scale systems.

The problem of having high detection rates (equivalently, few false negatives) and few false positives, is very familiar in the context of software verification. The plane manufacturer Airbus, for example, uses program-analysis techniques to detect possible memory leaks in on-board software. Since essentially all program-analysis problems are formally undecidable (meaning that no algorithm can compute accurate analysis information for all analyzed programs), only approximate solutions can be computed. Most analyses over-approximate, meaning that they catch all actual errors (zero false negatives) but also may report spurious ones. But if such an over-approximating analysis reports 900 potential memory leaks for a $100,000$ line computer program, the engineers may just abandon the use of this technique and resort to manual code inspection of what they perceive to be the critical parts of the program.

This suggests that the use of over-approximation with a low false-positive rate is attractive for the detection of insider threats. We want to catch all insider attacks, but we only want very few false alarms. The choice of the particular program analysis is informed by the program's "threat model". For example, checking that a program terminates is vital if its termination ensures the release of locks in the kernel of an operating system [9]. But for an application program, non-termination may be merely an inconvenience, whereas buffer overflows (detectable with another form of program analysis) may then compromise the entire machine on which the application runs. In short, the choice of analysis is dependent on context.

For any system that detects events that may signify an insider attack, we would expect a similar dependence on context and intent when formalizing what actually constitutes an insider attack. Risk and cost considerations, combined with the trustworthiness of users should shape these definitions. A detection system may only be concerned with worst-case threats, for example, and may not concern itself with petty insider abuse of systems. On the other hand, petty but persistent abuse may not merely be about misusing office equipment for private needs. It might indicate the

build-up of a slow but powerful insider attack. This suggests the need for the coordination of detection systems that each have their own intent and purpose and that need the ability to share their information in semantically meaningful ways across different intent boundaries.

4 Toward a context- and insider-aware policy language

We observed in the previous section that the insider problem arises because of a discrepancy between trust and trustworthiness. In the context of access control, we note that there are two points at which trust is of particular relevance.

First, a human user must authenticate to a computer, thereby generating some binding of the human user to security-related information, such as a user ID, security group identifiers, role identifiers, security labels, etc. We say that (successful) authentication gives rise to a *security context* that is associated with all programs executed by the authenticated user, and it is this context that will be used to evaluate whether subsequent user actions are authorized or not.

One method of authentication may be deemed to be more "secure" or "trustworthy" than other methods, so we can have greater confidence in the trustworthiness of the processes that are spawned as a result of a successful authentication event that employs this particular method. One natural approach would be to make the security context a function of both the user identity and the method of authentication, so that a "stronger" (whatever that might mean) authentication method means (ultimately) that the authenticated user will be authorized to perform a greater variety of actions. Of course, we must now consider what makes one method of authentication more secure than another. This may be determined by the way in which the user authenticates (password, biometric, knowledge of some secret key, etc.), or it may be determined by the user's location (local, remote, etc.), or indeed by many other possible factors.

The second point at which trust becomes relevant is when this security context is used to determine the actions for which the associated user is authorized. In many situations, the security context comprises identifiers, such as roles, that are directly associated with authorizations. (For our purposes, a *principal* is an entity or identifier that is directly authorized for certain actions by an authorization policy.) But the security context need not necessarily consist of principals: in Unix, for example, the security context (comprising security identifiers for user and group accounts) is mapped at request evaluation time to exactly one principal from the set $\{\texttt{owner}, \texttt{group}, \texttt{world}\}$.

With regard to this aspect of trust, we need to examine the justification for binding a user (or some part of a user's security context) to a principal, since it is this binding that ultimately decides the actions for which the user is authorized. This binding may be delayed, as in Unix, until request time. Unlike Unix, this binding may also be dependent on conditions and information not traditionally considered when evaluating access requests. (We may not wish to bind a user to certain princi-

pals if the location of the user is anomalous, for example.) A policy-based approach may well be useful here, and would be rather similar to existing *context-aware* access control systems.

We will concentrate on this approach in the remainder of this paper. In this section, we consider how we might develop a general-purpose language for specifying access-control policies that take factors such as risk and the trustworthiness of authorized users into account.

As we have said, we believe that the central problem is that trusted users may not be trustworthy. (That is, we need to be able to deal with a user for whom an erroneous, unjustified or obsolete assumption has been made about his reliability.) The question, then, is how to define access-control policies in such a way that the privileges for which users are authorized can be modified automatically (ie, without human intervention) when those users are revealed, or suspected, to be untrustworthy.[3] We therefore list additional requirements for addressing the insider problem within policy-based, access-control systems.

4.1 Context and request predicates

To understand these requirements, it is useful to think of access requests as tuples of the form (s,o,a,c) where s ranges over subjects (*i.e.*, user proxies), o ranges over objects (*i.e.*, protected resources), a ranges over actions (such as *read*, *write*, etc.), and c ranges over contextual information (such as whether or not the requestor is accessing the system remotely). Sets of such tuples are thus subsets X of a space $S \times O \times A \times C$ of access requests. Such subsets can therefore be interpreted as predicates, which we refer to as *request predicates*. Such predicates may depend only on context, so they could be seen as describing subsets of C. We then call such predicates *context predicates*.

Given $X \subseteq S \times O \times A \times C$, we write p_X^+ to denote the policy that authorizes all requests in X and denies all others. [4] Dually, policy p_X^- authorizes a request if and only if it is not in X.

4.2 Requirements

Then our requirements for policy-based access control that can address insider threats can be described as follows:

[3] Of course, a preliminary problem is how to reliably identify and report to the access-control system those users whose trustworthiness is questionable. We do not consider this problem here.

[4] Strictly speaking, it is the policy-decision point that decides whether or not a request is authorized by a policy or not. However, we don't make this distinction explicit henceforth.

1. The ability to support richer types of policy decisions beyond the usual binary "deny" and "grant", so that inconsistencies, definitional gaps, error conditions, etc., can be articulated, accommodated, and propagated.
2. The ability to declare context predicates and request predicates, where the latter express sets of access requests of interest and the former capture contextual state; and the ability to transform such declarations into access-control policies, for example by turning predicates X into policies p_X^+ or p_X^-.
3. The ability to transform an access-control policy p into a set of access requests X that captures some aspect of the behavior of policy p. For example, we might want to identify all requests authorized by policy p in a specific context. However, this is by no means the only possibility. We may, for example, wish to compute those requests for which the policy p is over-defined (conflicts) or under-defined (gaps).
4. Support for typed, modular programming of access-control policies, with a rich set of declarative coordination patterns. This would facilitate the creation of robust, composed authorization policies and would leverage the abilities of the previous two requirements.

4.3 Policy transformations via declarative programming

Given these requirements, we propose that there is value in transferring the principles of declarative programming into the domain of context- and trustworthiness-aware access control. Declarative programming (by which we loosely mean extensions of functional programming, logic programming; or modeling languages based on logic such as Alloy [14]) focuses on the "what" and not on the "how" of computation. So declarative access-control policies are concerned with what requests will be granted under which contextual circumstances, but they don't have to specify how such policies are then enforced.[5]

This separation of concerns between policy specification and implementation brings additional benefits. One can design declarative languages that have typed modules and so support robust policy composition that facilitates reuse and offers a policy authoring tool that is hopefully descriptive and intuitive enough for policy writers.[6] Indeed, we believe that it is this support of modular, declarative programming that will be able to accommodate context- and trustworthiness-aware access control that has a flexible range of granularity and is not necessarily identity-based.

[5] We assume that declarations will have intuitive operational or denotational semantics and will therefore be implementable.

[6] It is of interest to note that the financial trading community has now recognized the need to be able to program as close to the user domain (*e.g.*, financial traders) as possible. Declarative programming, mostly in the form of functional programming, is becoming an important tool in that sector, even for back-office aspects such as specifying and analyzing contracts [15].

4.4 Discussion of requirements

We now discuss each of these requirements in more detail and illustrate what their realizations would and will provide. The first requirement allows us to treat a wide variety of functions and inputs as "policies". For example, a policy might be composed of sub-policies that return quantitative decisions, *e.g.*, about trustworthiness or risk, and these non-binary decisions could then be converted into appropriate binary policy decisions.

The second requirement suggests the use of abstract request predicates and propositional-logic connectives so that one can form expressions such as

$$\texttt{Manager} \wedge \texttt{OnDuty} \wedge \neg \texttt{Weekend} \tag{1}$$

The access request is left implicit. Indeed, an abstract request predicate may not even depend on the access request. For example, if a subject is requesting to edit a certain PDF document, the evaluation of the above expression will not depend on the type of action or type of document, but will depend on whether said subject presently has role `Manager`, is currently on duty, and whether the request occurs not during a week-end. So `Weekend` would better be seen as a context predicate.

The expression in (1) is declarative since it does not elaborate on how the connections between, say, `Manager` and subjects are being implemented. In particular, it does not necessarily commit to an underlying RBAC model nor explicitly depend on the handling of secure attributes, etc. The provision of abstract interfaces for specifying access-control policies was suggested and developed in the work of Bruns & Huth in [6, 5] already.

One can now promote expressions such as the one in (1) to genuine policies, *e.g.*,

$$\texttt{Grant if Manager} \wedge \texttt{OnDuty} \wedge \neg \texttt{Weekend} \tag{2}$$

which is the declarative way of encoding policies of form p_X^+ discussed above. We adopt here the interpretation of [5, 6], where the policy in (2) grants all requests that satisfy the composed predicate in (1), and where said policy has a gap (meaning it is undefined) at all other requests.

The third requirement allows us to identify sets of access requests that stem from policies themselves. For sake of illustration, consider two primitives

$$p\texttt{@Denies} \qquad\qquad p\texttt{@Grants}$$

that, given an access-control policy p, declare request predicates that capture that set of access requests that policy p denies, respectively grants. It should be noted that policy p may well not be equal to policy `Grant if` p`@Grants`. The policy `Grant if` p`@Grants` only has two possible decisions: grants and gaps. But policy p may have a number of possible decisions, such as conflicts or errors.

Meeting the fourth requirement means that we can wrap policy composition patterns into parameterized modules, and then instantiate or reuse such modules in policy composition or orchestration. For example,

```
pol insiderThreat(abnormalBeh : reqs, insider: reqs) {
  (grant if !abnormalBeh & insider) >
  (deny  if abnormalBeh | !insider)
}
```

declares such a module with two types, `pol` for policies and `reqs` for request predicates. This module takes as input two request predicates and outputs a policy, where > declares a priority composition. The output policy therefore grants a request if it comes from an insider and if no abnormal behavior has been flagged; and it denies a request if either abnormal behavior has been flagged or the request does not come from an insider. The module itself does not state how insiders and abnormal behavior are defined. But this separation of concern creates flexibility, since different circumstances may require different notions of insiders or abnormality.

For example, `abnormalBeh` could indicate whether a request wants to copy unusually large amounts of data from one medium or location to another. Or `abnormalBeh` could encapsulate a risk-based or statistical analysis of the requestor's behavior, and so reduce such quantitative results to a binary authorization "decision".

In another context, `insider` may declare those sets of requests in which either a manager assigns pay increments to non-managers or in which non-managers rate the performance of their managers, as in

$$\begin{aligned}\texttt{insider} = {} & (\texttt{manager} \wedge \texttt{assign} \wedge \texttt{payIncrease}) \vee \\ & (\neg\texttt{manager} \wedge \texttt{rate} \wedge \texttt{managerPerformance})\end{aligned} \tag{3}$$

We point out that requestors that satisfy the predicate `insider` defined above are not defined in mere terms of roles, but in terms of the combination of role, object, and action. In particular, managers are considered as "outsiders" (*i.e.*, not authorized or trusted) when it comes to rating the performance of managers. Of course, such combinations could be enriched with additional context predicates.

4.5 Policy transformations

Request predicates also leverage policy analysis. Since p@`Grants` and `insider` are both request predicates, we can examine whether the implication

$$p@\texttt{Grants} \rightarrow \texttt{insider} \tag{4}$$

is valid. If so, then policy p will not grant requests unless they stem from those who are declared to be insiders. This would show that policy p cannot be per-

suaded to grant access to outsiders, assuming that outsiders are implicitly defined by ¬insider.

This approach does not impose the exclusive use of static-analysis methods though. A dynamic version of (4) would test its validity at run-time, and adapt policy behavior accordingly, if needed. We can express this in a parameterized module:

```
pol grantOnlyInsiders(P : pol, insider : reqs) {
  (grant if P@grants & insider) > deny
}
```

The idea here is that the module takes a policy P and a request predicate insider as input, and retrofits policy P such that the retrofitted policy will grant only if the request is in the "set" insider of interest and the policy P would grant that request. The retrofitted policy denies all other requests, including those that don't satisfy insider but would be granted by P. This ensures that (4) holds in all executions of the access-control system, but now for the invocation

```
grantOnlyInsiders(P,insider)
```

instead of for P.

Finally, we give a flavor of how policy languages that meet our requirements could realize the approach we had suggested in Section 3, which uses trustworthiness to inform a risk-evaluation strategy for access control.

4.6 Risk- and trustworthiness-aware policy composition

Our first requirement stated that policy decisions need not be binary, and could even be continuous. We also argued that trustworthiness is an appropriate and useful notion for dealing with the insider problem. We now illustrate how these things can be brought together within our policy language: The parameterized module

```
bool tooRisky(R: req; riskThreshold : double) {
  return (cost(R) / trustworthiness(R.subject)
         > riskThreshold)
}
```

returns a Boolean and takes as input single request R and a threshold riskThreshold for the risk the access-control system is willing to accept when granting this request. The risk is perceived as being too high if the estimated cost (that will be incurred if

the request is authorized and subsequently abused) divided by the requestor's trustworthiness (the probability of the requestor not abusing the authorization) is strictly above a specified risk threshold.

We illustrate how this function could be used to make a policy risk-aware. For example

```
pol riskFilter(P : pol) {
  (deny if tooRisky(this)) > P
}
```

is a module that takes policy `P` as input and then produces a policy that denies if the presented request `this` is judged to be too risky; otherwise, it will behave as `P`.

We will now briefly consider the architectural ramifications of supporting the evaluation policies written in our context- and trustworthiness-aware policy language.

5 Access-control architectures and the insider problem

In considering an architecture that accommodates the awareness of risk and trustworthiness, there are good reasons for reusing existing access-control architectures such as the standard one depicted in Figure 1. For one, it may simply be infeasible to rebuild an entire access-control system and so one might be committed to its underlying architecture. For another, adapting an existing and already implemented architecture may be much more cost-effective. Moreover, such reuse readily accommodates legacy systems that don't yet support or indeed don't require support for such notions of awareness.

Assuming we adopt the core architecture shown in Figure 1, we need to add hardware and software components that can provide the measures of trustworthiness, the observations of context, and the assessment of risk. In particular, we will need the following:

1. trust-management software for evaluating user trustworthiness
2. event-monitoring systems (both hardware and software), including actuators and sensory networks (if applicable)
3. risk-evaluation systems (that may combine stochastic with worst-case uncertainty)
4. auditing systems and audit-processing software

Existing research in each of these areas will certainly be of use here, as well as recent work on usage control in access control [21] and on intrusion-prevention systems [17].

The generic architecture modified in this manner is depicted in Fig. 2. What is not shown in detail, though, are the necessary programming interfaces for determining the values of context and request predicates that occur in the access-control policy. But our language viewed such predicates already as abstract interfaces, *i.e.*, as methods with an implicit parameter (a contextual request that we exposed with construct `this` in the body of method `riskFilter` above). Request predicate `manager`, *e.g.*, could be seen as a method that returns a set of requests:

```
reqs manager() { implementation code }
```

Similarly, as we have already demonstrated, quantitative predicates based on risk and trustworthiness are written as parameterized methods. So at the policy-decision point the policy can successfully evaluate all these methods provided that they have been implemented in the policy repository.

6 Concluding remarks

The insider problem is extremely difficult to define, much less to solve. In this chapter we have identified the contribution that access control could make to the insider problem in general, and proposed a policy-based framework for such access control. Our key observation was that problems of insiderness could be rephrased in terms of problems of trust and trustworthiness.

Based on that observation, we then demonstrated how principles from declarative programming could be combined with established policy-composition principles to obtain a policy language in which the management and coordination of risk, trustworthiness, and conventional request processing could be accommodated. We illustrated how such coordinations could support the prevention or mitigation of insider attacks – by dynamically adjusting levels of trustworthiness on the bases of information received from the context handlers of the access-control system. In future work we mean to examine existing case studies of insider attacks to determine whether our approach could have detected, prevented or otherwise mitigated such attacks if policies had been made risk-aware.

We conclude by discussing related work that is relevant for extending the approach that we have described in this chapter.

Bruns and Huth have shown that primitives such as p@`Grants` and p@`Denies` can be defined as request predicates if policies are composed out of request predicates with operators similar to those discussed in this chapter [5]. In their work, request predicates were atomic, without structure. But it is easy to extend policy composition and the above primitives to request predicates that support composition operators from propositional logic, as demonstrated in the talk [13] at the Dagstuhl Seminar [2].

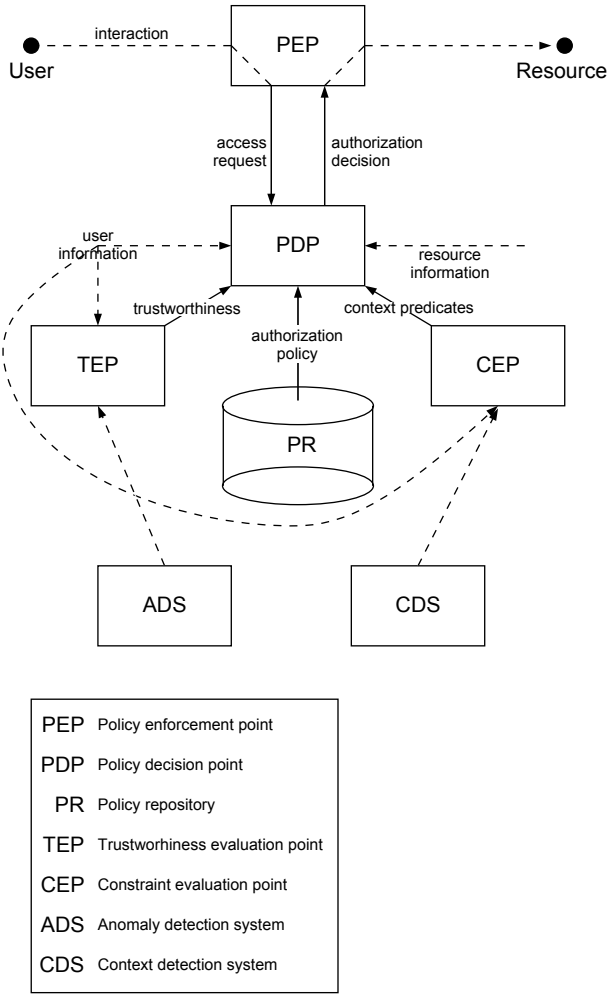

Fig. 2 An extended access-control architecture that can provide support for risk- and trustworthiness-aware access control

The ideas put forward in this chapter are perhaps closest to the work by Probst *et al.* in [24]. The authors point out that the predominant approach to addressing the insider problem is to maintain audit logs for "post mortem" analysis. They suggest a formalism for high-level access-control models, where models can then be mapped into terms of a process algebra. Those terms are then subjected to static analyses that yield safe overapproximations of users' capabilities and restrictions. Our proposal extracts expressions that state such capabilities and restrictions, but we can not only

use them for static analysis but also as wrappers that can extend or restrict given policies at run-time.

References

1. Bishop, M., S. Engle, S. Peisert, S. Whalen, and C. Gates, *Case Studies of an Insider Framework*, Proc. of Hawaii International Conference on System Sciences, pp. 1–10, IEEE Computer Society Press, 2009.
2. Bishop, M., D. Gollmann, J. Hunker, and C. W. Probst, *Countering Insider Threats*, Dagstuhl Seminar 08302, Leibnitz Center for Informatics, 18 pp., Dagstuhl Seminar Proceedings, ISSN 1862 - 4405, July 2008.
3. Bishop, M., *Panel: The Insider Problem Revisited*, Proc. of NSPW 2005, ACM Press, 2006.
4. Brackney, R., and R. Anderson, *Understanding the Insider Threat*, Proc. of a March 2004 Workshop, RAND Corp., Santa Monica, California, March 2004.
5. Bruns, G., and M. Huth, *Access-Control Policies via Belnap Logic: Effective and Efficient Composition and Analysis*, Proc. of CSF 2008, pp. 163–178, IEEE Computer Society Press, 2008.
6. Bruns, G., D. S. Dantas, and M. Huth, *A simple and expressive semantic framework for policy composition in access control*, Proc. of FMSE 2007, pp. 12–21, ACM Press, 2007.
7. Chakraborty, S. and I. Ray, *TrustBAC: integrating trust relationships into the RBAC model for access control in open systems*, Proc. of SACMAT '06, pp. 49–58, ACM Press, 2006.
8. Cheng, P.-C., P. Rohatgi, C. Keser, P. A. Karger, and G. M. Wagner, *Fuzzy Multi-Level Security: An Experiment on Quantified Risk-Adaptive Access Control*, IBM Research Report, RC24190 (W0702-085), Computer Science, February 2007.
9. Cook, B., A. Podelski, and A. Rybalchenko, *Terminator: Beyond safety*, Proc. of CAV'06, LNCS 4144, pp. 415–418. Springer, (2006.
10. Cranor, L. F. and S. Garfinkel (editors), *Security and Usability – Designing Secure Systems That People Can Use*, O'Reilly, California, August 2005.
11. *Department of Defense Trusted Computer System Evaluation Criteria*, Technical Report DoD 5200.28-STD, US Department of Defense, 1985.
12. Hoffman, K., D. Zage, and C. Nita-Rotaru, *A Survey of Attack and Defense Techniques for Reputation Systems*, To appear in ACM Computing Surveys, Volume 41, Issue 4, December 2009.
13. Huth, M., *A Simple Language for Policy Composition and Analysis*, Talk given at [2]. `www.doc.ic.ac.uk/~mrh/talks/Dagstuhl08.pdf`
14. Jackson, D., *Software Abstractions: Logic, Language, and Analysis*, MIT Press, 2006.
15. Jones, S. P., J.-M. Eber, and J. Seward, *Composing contracts: an adventure in financial engineering (functional pearl)*, ACM SIGPLAN Notices 35(9): 280–292, ACM Press, 2000.
16. Lee, A. and T. Yu, *Towards a dynamic and composable model of trust*, Proc. of SACMAT'09, pp. 217–226, ACM Press.
17. Locasto, M. E., K. Wang, A. D. Keromytis, and S. J. Stolfo, *FLIPS: Hybrid Adaptive Intrusion Prevention* , in: Recent Advances in Intrusion Detection, LNCS 3858, pp. 82–101, Springer, 2006.
18. Moore, A. P., D. M. Cappelli, and R. F. Trzeciak, *The "Big Picture" of Insider IT Sabotage Across U.S. Critical Infrastructures*, Technical Report CMU/SEI-2008-TR-009, ESC-TR-2008-009, Carnegie Mellon University, May 2008.
19. The New York Times, *French Bank Says Rogue Trader Lost $7 Billion* , 25 January, 2008.
20. Patzakis, J., *New Incident Response Best Practice: Patch and Proceed is No Longer Acceptable Incident Response Procedure*, Guidance Software, Pasadena, California, September 2003.
21. Park, J., and R. S. Sandhu, *The $UCON_{ABC}$ usage control model*, ACM Trans. Inf. Syst. Secur. 7(1): 128–174, ACM Press, 2004.

22. Park, J. S. and J. Giordano, *Role-Based Profile Analysis for Scalable and Accurate Insider-Anomaly Detection*, Proc. IPCCC'06, 2006.
23. Park, J. S. and J. Giordano, *Access Control Requirements for Preventing Insider Threats* , Proc. ISI'06 LNCS 3975, pp. 529–534, Springer, 2006.
24. Probst, Ch. W., R. R. Hansen, and F. Nielson, *Where Can an Insider Attack?*, Proc. of FAST'06, LNCS 4691, pp. 127–142, Springer, 2006.
25. Probst, Ch. W. and J. Hunker, *The Risk of Risk Analysis-And its relation to the Economics of Insider Threats*, Proc. of the Eighth Workshop on the Economics of Information Security (WEIS 2009), June 2009.
26. Sandhu, R. S., E. J. Coyne, H. L. Feinstein, and C. E. Youman, *Role-Based Access Control Models*, IEEE Computer 29(2): 38–47, 1996.
27. Viega, J. and G. McGraw, *Building Secure Software*, Addison-Wesley Professional Computing Series, 2002.

Monitoring Technologies for Mitigating Insider Threats

Brian M. Bowen, Malek Ben Salem, Angelos D. Keromytis, and Salvatore J. Stolfo

Abstract In this chapter, we propose a design for an insider threat detection system that combines an array of complementary techniques that aims to detect evasive adversaries. We are motivated by real world incidents and our experience with building isolated detectors: such standalone mechanisms are often easily identified and avoided by malefactors. Our work-in-progress combines host-based user-event monitoring sensors with trap-based decoys and remote network detectors to track and correlate insider activity. We introduce and formalize a number of properties of decoys as a guide to design trap-based defenses to increase the likelihood of detecting an insider attack. We identify several challenges in scaling up, deploying, and validating our architecture in real environments.

1 Introduction

The annual Computer Crime and Security Survey for 2008 [13] surveyed 522 security employees from US corporations and government agencies, finding that insider incidents were cited by 44 percent of respondents, nearly as high as the 49 percent

Brian M. Bowen
Department of Computer Science, Columbia University, 1214 Amsterdam Ave MC 0401, New York, NY 10027, e-mail: bmbowen@cs.columbia.edu

Malek Ben Salem
Department of Computer Science, Columbia University, 1214 Amsterdam Ave MC 0401, New York, NY 10027, e-mail: malek@cs.columbia.edu

Angelos D. Keromytis
Department of Computer Science, Columbia University, 1214 Amsterdam Ave MC 0401, New York, NY 10027, e-mail: angelos@cs.columbia.edu

Salvatore J. Stolfo
Department of Computer Science, Columbia University, 1214 Amsterdam Ave MC 0401, New York, NY 10027, e-mail: sal@cs.columbia.edu

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*
DOI 10.1007/978-1-4419-7133-3_9,

that encountered a conventional virus in the previous year. In general, there is an increasing recognition of the significance, scope and cost of the malicious insider problem. Some state-of-the-art defenses focus on forensics analysis and attribution after an attack has occurred using techniques such as sophisticated auditing [20] and screen capture [12]. Other commercially available systems are designed to prevent, detect, and deter insider attack. The ideal case is to devise systems that *prevent* insider attack. Policy-based mechanisms and access control systems have been the subject of study for quite some time but have not succeeded in solving the problem of preventing insider abuse. Monitoring, detection, and mitigation technologies are realistic necessities.

Detection systems have been designed to identify specific attack patterns or deviations from known, long-term user behavior. Such techniques are typically used as part of standalone mechanisms rather than in an integrated defense architecture. For that reason, the malicious behavior detection-based insider defenses suffer from several problems.

- Since behavior is a noisy approximate of user intent in the absence of sufficient contextual information about the user and the overall environment, such systems are typically tuned to minimize false alerts by being less stringent about what is considered malicious. While reducing the administrators' workload and the users' irritation factor, such tuning may allow some malicious behavior to go undetected. As with many probabilistic detection mechanisms, the tradeoffs inherent in tuning systems are poorly understood.
- Since the relationship between behavior and intent is hard to determine, and alarms may be false, it is difficult to confidently take some action (whether automated or at the human level) in response to an alert.
- By operating as standalone mechanisms, it is easy for an adversary with some knowledge of their existence to either evade or even *disable* them. In fact, we see an increasing number of malware attacks that disable defenses such as anti-virus software and host sensors prior to undertaking some malicious activity [5].

To address the malicious insider problem, we posit that systems must leverage multiple complementary and mutually supportive techniques to detect and deter intentionally malicious adversaries. We direct our efforts against inside attackers that have some, but perhaps not complete knowledge of the enterprise environment. In this chapter, we do not address the important problem of malicious system administrators who have control over all defensive systems. This remains a particularly interesting open problem.

The first component in our architecture is a decoy document generation system to deceive an insider by leveraging uncertainty of the authenticity of information that may be accessed in an unauthorized manner. Our system generates realistic-looking documents that contain both decoy credentials that are monitored for (mis)use, and stealthy embedded beacons that signal when the document is opened. Beacons are embedded in documents using methods of deception and obfuscation gleaned from studying malcode embedded in malicious documents as seen in the wild [9].

The network component integrates monitored network traps with the decoy document generation component, and allows our system to isolate the activity of malicious users. These traps allow us to follow "personalized" decoy credentials even after they leave the local environment. As with honeypots [16], the misuse of the network traps guarantees an insider has misappropriated internal information. The absconded information may even have been accessed in a way that has evaded detection (and even forensic analysis) altogether. Nonetheless, our system can determine the time and location of the leak through the use of decoys embedded in the content of the decoy document, allowing further forensic investigation and damage assessment.

The host-based sensors we designed in our architecture, collectively named "Are You You?" (RUU), collect low-level audit data from which we identify specific user actions. RUU profiles user actions to form a baseline of normal behavior utilizing anomaly detection techniques to isolate behavior differences over time. Subsequent monitoring for abnormal behaviors that exhibit large deviations from this baseline signal a potential insider attack. We conjecture that the evidence provided by the host sensor combined with other detection techniques will indicate insider activity that violates policy. It is often noted that, on their own, anomaly detection systems have high levels of false positives. Combining multiple views of the same event can dramatically reduce the number of false positives associated with a malicious event [8].

In this chapter we:

- introduce and analyze a system that uses multiple sensors in a scalable fashion to leverage context for detecting malicious insider activity;
- describe new lightweight sensors that model user actions on the host level;
- present a design for decoys that combines a number of methods and monitors, both internal and external to an organization, to detect insider exploitation using ordinary-looking documents as bait;
- introduce a set of generally applicable properties to guide the design and deployment of decoys and maximize the deception they induce for different classes of insiders who vary by their level of knowledge and sophistication;
- present a large-scale automated decoy creation and management system for deploying baited documents that allow us to detect the presence (and, in some cases, identity) of malicious insiders, or at least determine the existence of malicious insider activity. This provides a means for ordinary users to deploy decoys on their hard drives without having to deploy and configure sophisticated honeypots and sensors. Users are alerted by email when a decoy has been touched on their laptops and personal computers; and
- present a preliminary analysis of the overhead and effectiveness of the system in a realistic environment.

2 Related Research

The use of deception, or decoys, plays a valuable role in the protection of systems, networks, and information. The first use of decoys (*i.e.*, in the cyber domain) has been credited to Cliff Stoll [23, 17] and detailed in his novel "The Cuckoos Egg" [18], where he provides a thorough account of his crusade to catch German hackers breaking into Lawrence Berkeley Laboratory computer systems. Stoll's methods included the use of bogus networks, systems, and documents to gather intelligence on the German attackers who were apparently seeking state secrets. Among the many techniques waged, he crafted "bait" files, or in his case, bogus classified documents that really contained non-sensitive government information and attached "alarms" to them so that he would know if anyone accessed them. To Stoll's credit, a German hacker was eventually caught and it was found that he had been selling secrets to the KGB.

Honeypots are effective tools for profiling attacker behavior. Honeypots are considered to have low false positive rates since they are designed to capture only malicious attackers, except for perhaps an occasional mistake by innocent users. Spitzner described how honeypots can be useful for detecting insider attack [16] and discussed the use of honeytokens [17] such as bogus medical records, credit card numbers, and credentials. In a similar spirit, Webb *et al.* [21] showed how honeypots can be useful for detecting spammers. In current systems, the decoy/honeytoken creation is a laborious and manual process requiring large amounts of administrator intervention. Our work extends these basic ideas to an automated system of managing the creation and deployment of these honeytokens.

Yuill *et al.* [23] extend the notion of honeytokens with a "honeyfile system" to support the creation of bait files, or as they define them, "honeyfiles." The honeyfile system is implemented as an enhancement to the Network File Server. The system allows for any file within user file space to become a honeyfile through the creation of a record associating a filename to userid. The honeyfile system monitors all file access on the server and alerts users when honeyfiles have been accessed. This work does not focus on the content or automatic creation of files, but does mention some of the challenges in creating deceptive files (with respect to names) that we address as well.

In this chapter, we introduce a set of properties of decoys to guide their design and maximize the deception which they induce for different classes of insiders with various levels of knowledge and sophistication. To the best of our knowledge, the synthesis of these properties is novel. Bell and Whaley [1] described the structure of deception as a process of hiding the real and showing the false. They introduced several methods of hiding that include masking, repackaging, and dazzling, along with three methods of showing that include mimicking, inventing, and decoying. Yuill *et al.* [22] expand upon this work and characterize deceptive hiding in terms of how it defeats an adversary's discovery process. They describe an adversary's discovery process as taking three forms: direct observation, investigation based on evidence, and learning from other people or agents. Their work offers a process model for cre-

ating deceptive hiding techniques based on how they defeat an adversary's discovery process.

3 Threat Model - Level of Sophistication of the Attacker

The Decoy Document generation system described in Section 5 relies on the use of decoys to deceive, confuse, and confound attackers, ultimately forcing them to expend far more effort to discern real information from bogus information. To understand the capability of the various decoys that we introduce, it is necessary to first explore the various levels of attacker sophistication. We broadly define four monotonically increasing levels of insider sophistication and capability that may be used to break through the deception our decoys seek to induce. Some will have tools available to assist in deciding what is a decoy and what is real. Others will only have their own observations and insights.

Low: Direct observation is the only tool available. The adversary largely depends on what can be gleaned from a first glance. We strive to defeat this level of adversary with our beacon documents, even though decoys with embedded beacons may be distinguished with more advanced tools.
Medium: A more thorough investigation can be performed by the insider; Decisions based on other information can be made. For example, if a decoy document contains a decoy account credential for a particular identity, an adversary may verify that the particular identity is real or not by querying an external system (such as `http://www.whitepages.com/`). Such adversaries will require stronger decoy information, possibly corroborated by other sources of evidence.
High: Access to the most sophisticated tools is available to the attacker (*e.g.*, super computers, other informed people who have organizational information). The notion of the "Perfect Decoy" described in the next section may be the only indiscernible decoy by an adversary of such caliber.
Highly Privileged: Probably the most dangerous of all is the privileged and highly sophisticated user. Such attackers will be fully aware that the system is baited and will employ sophisticated tools to try to analyze, disable, and avoid decoys entirely. As an example of how defeating this level of threat might be possible, consider the analogy with someone who knows encryption is used (and which encryption algorithm is used), but still cannot break the system because they do not have knowledge of an easy-to-change operational parameter (the key). Likewise, just because someone knows that decoys are used in the system does not mean they should be able to identify them all. Devising a hard-to-defeat scheme is the principle we explore in the next section.

We further define insider threats by differentiating between *Masqueraders* (attackers who impersonate another system user) and *Traitors* (attackers using their own legitimate system credentials) who each have varying levels of knowledge. The masquerader is presumed to have less knowledge of a system than the victim user

whose credentials were stolen. The *innocent insider* who mistakenly violates policy is undoubtedly the largest population of insiders that we also target using trap-based decoys.

4 Decoy Properties

In order to create decoys to bait various levels of insiders, one must understand the core properties of a decoy that will successfully bait an insider. We enumerate various properties and means of measuring these properties that are associated with decoy documents to ensure their use will be likely to snare an inside attacker. We introduce the following notation for these definitions.

Believable[1]: Capable of eliciting belief or trust; capable of being believed; appearing true; seeming to be true or authentic.

A good decoy should make it difficult for an adversary to discern whether they are looking at an authentic document from a legitimate source or if they are indeed looking at a decoy. We conjecture that believability of any particular decoy can be measured by adversary's failure to discern one from the other. We formalize this by defining a decoy believability experiment. The experiment is defined for the document space M with the set of decoys D such that $D \subseteq M$ and $M - D$ is the set of authentic documents.

The Decoy Believability Experiment: $\mathbf{Exp}^{believe}_{A,D,M}$

- For any $d \in D$, choose two documents $m_0, m_1 \in M$ such that $m_0 = d$ or $m_1 = d$, and $m_0 \neq m_1$; that is, one is a decoy we wish to measure the believability of and the second is chosen at random from the set of authentic documents.
- Adversary A obtains m_0, m_1 and attempts to choose $\hat{m} \in \{m_0, m_1\}$ such that $\hat{m} \neq d$, using only information intrinsic to m_0, m_1.
- The output of the experiment is 1 if $\hat{m} \neq d$ and 0 otherwise.

For concreteness, we build upon the definition of "Perfect Secrecy" proposed in the cryptography community [7] and define a "perfect decoy" when:

$$\Pr[\mathrm{Exp}^{believe}_{A,D,M} = 1] = 1/2$$

The decoy is chosen in a believability experiment with a probability of 1/2 (the outcome that would be achieved if the volunteer decided completely at random). That is, a perfect decoy is one that is completely indistinguishable from one that is not. A benefit of this definition is that the challenge of showing a decoy to be believable, or not, reduces to the problem of creating a "distinguisher" that can decide with probability better than 1/2.

[1] For clarity, each property is provided with its definition gleaned from online dictionary sources.

In practice, the construction of a "perfect decoy" might be unachievable, especially through automatic means, but the notion remains important as it provides a goal to strive for in our design and implementation of systems. For many threat models, it might suffice to have less than perfect believable decoys. For our proof-of-concept system described below, we generate receipts and tax documents, and other common form-based documents with decoy credentials, realistic names, addresses and logins, all information that is familiar to all users.

We note that the believable property of a decoy may be less important than other properties defined below since the attacker may have to open the decoy in order to decide whether the document is real or not. The act of opening the document may be all that we need to trap the insider, irrespective of the believability of its content. Hence, enticing an attacker to open a document, say one with a very interesting name, may be a more effective strategy to detect an inside attack than producing a decoy document with believable content.

Enticing: highly attractive and able to arouse hope or desire; "an alluring prospect"; lure.

Herein lies the issue of how does one measure the extent to which a decoy arouses desires, how well is it a lure? One obvious way is to create decoys containing information with monetary value, such as passwords or credit card numbers that have black market value [19]. However, enticement depends upon the attacker's intent or preference. We define enticing documents in terms of the likelihood of an adversary's preference; enticing decoys are those decoys that are chosen with the same likelihood. More formally, for the document space M, let P be the set of documents of an adversary's A preference, where $P \subseteq M$. For some value ε such that $\varepsilon > 1/|M|$, an enticing document is defined by the probability

$$\Pr[m \rightarrow M | m \in P] > \varepsilon$$

where $m \rightarrow M$ denotes m is chosen from M. An enticing decoy is then defined for the set of decoys D, where $D \subseteq M$, such that

$$\Pr[m \rightarrow M | m \in P] = \Pr[d \rightarrow M | d \in D]$$

We posit that by defining several general categories of "things" that are of "attacker interest", one may compose decoys using terms or words that correspond to desires of the attacker that are overwhelmingly enticing. For example, if the attacker desires money, any document that mentions or describes information that provides access to money should be highly enticing. We believe we can measure frequently occurring (search) terms associated with major categories of interest (*e.g.*, words or terms drawn from finance, medical information, intellectual property) and use these as the constituent words in decoy documents. To measure the effectiveness of this generative strategy, it should be possible to execute content searches and count the number of times decoys appear in the top 10 list of displayed documents. This is a reasonable approach also, to measuring how conspicuous, defined below, the decoys become based upon the attacker's searches associated with their interest and intent.

Conspicuous: easily visible; easily or clearly visible; obvious to the eye or mind; Attracting attention.

A *conspicuous* decoy should be easily found or observed. Conspicuous is defined similar to enticing, but conspicuous documents are found because they are easily observed, whereas enticing documents are chosen because they are of interest to an attacker. For the document space M, let V be the set of documents defined by the minimum number of user actions required to enable their view. We use a subscript to denote the number of user actions required to view some set of documents. For example, documents that are in view at login or on the desktop (requiring zero user actions) are labeled V_0, those requiring one user action are V_1, etc. We define a "view", V_i of a set of documents as a function of a number of user actions applied to a prior view, V_{i-1}, hence

$$V_i = \text{Action}(V_{i-1}) \text{ where } V_j \neq V_i, j < i$$

An "Action" may be any command or function that displays files and documents, such as 'ls', 'dir', 'search.' For some value ε such that $\varepsilon > 0$, a conspicuous document, d, is defined by the probability

$$\prod_{i=0}^{n} \Pr[V_i] > \varepsilon$$

where n is the minimum value where $d \in V_n$. Note if d is on the desktop, V_0, $\Pr[V_0] = 1$ (*i.e.*, the documents in full view are highly conspicuous).

When a user first logs in, a conspicuous decoy should either be in full view on the desktop, or viewable after one (targeted) search action. One simple user action is optimal for a highly conspicuous decoy. Thus, a measure of conspicuousness may be a count of the number of search actions needed, on average, for a decoy to appear in full view. The decoy may be stored in the file system anywhere if a simple content-based search locates it in one step. But, this search act depends upon the query executed by the user. The query can either be a location (*e.g.*, search for a directory named "TAX" in which the decoy appears) or a content query (*e.g.*, using Google Desktop Search for documents containing the word "TAX.") In either case, if a decoy document appears after one such search, it is conspicuous. Hence, we may define the set P as all such files that can be found in some number of steps. But, this depends upon what search terms the attacker uses to query! If the decoy never appears because the attacker used the wrong search terms, the decoy is not conspicuous. We posit that the property of *enticing* is likely the most important property, and a formal measure to evaluate enticement will generate better decoys. In summary, an enticing decoy should be conspicuous to be an effective decoy trap.

Detectable; to discover or catch (a person) in the performance of some act: to detect someone cheating.

Decoys must ensure an alert is generated if they are exploited. Formally, this is defined for adversary A, document space M, and the set of decoys D such that

$D \subseteq M$. We use $Alert_{A,d} = 1$ to denote an alert for $d \in D$. We say d is detectable with probability ε when

$$\Pr[d \rightarrow M : Alert_{A,d} = 1] \geq \varepsilon$$

Ideally, ε should be 1.

We designed the decoy documents with several techniques to provide a good chance of detecting the malfeasance of an inside attack in real-time.

- At time of application start-up, the decoy document emits a beacon alert to a remote server.
- At the time of memory load, a host-sensor, such as an anti-virus scanner, may detect embedded tokens placed in a clandestine location of the document file format.
- At the time of ex-filtration, a NIDS such as Snort, or a stream event detection system such as Cayuga [4] may be used to detect these embedded tokens during the egress of the decoy document in network traffic where possible.
- At time of information exploitation and/or credential misuse, monitoring of decoy logins and other credentials embedded in the document content by external systems will generate an alert that is correlated with the decoy document in which the credential was placed.

This extensive set of monitors maximizes ε, forcing the attacker to expend considerable effort to avoid detection, and hopefully will serve as a deterrent to reduce internal malfeasance within organizations that deploy such a trap-based defense. In the proof-of-concept implementation reported in this paper, we focus our evaluation on the last item. We utilize monitors at our local IT systems, at Gmail and at an external bank.

Variability: The range of possible outcomes of a given situation; the quality of being subject to variation.

Attackers are humans with insider knowledge, even possibly with the knowledge that decoys are liberally spread throughout an enterprise. Their task is to identify the real documents from the potentially large cache of decoys. One important property of the set of decoys is that they are not easily identifiable due to some common invariant information they all share. A single search or test function would thus easily distinguish the real from the fake. The decoys thus must be highly varied. We define variable in terms of the likelihood of being able to decide the believability of a decoy given *any* known decoy. Formally, we define *perfectly variable* for document space M with the set of decoys D such that $D \subseteq M$ where

$$\Pr[d' \rightarrow D : \mathrm{Exp}^{believe}_{A,D,M,d'} = 1] = 1/2$$

Observe that, under this definition, an adversary may have access to *all* N previously generated decoys with the knowledge they are bogus, but still lack the ability to discern the N+1st. From a statistical perspective, each decoy is independent and

identically distributed. For the case that an adversary can determine the N+1st decoy only after observing the N prior decoys, we define this as an *N-strong Variant*.

Clearly, a good decoy generator should produce an unbounded collection of enticing, conspicuous, but distinct and variable documents. They are distinct with respect to string content. If the same sentence appears in 100 decoys, one would not consider such decoys with repetitive information as highly variable; the common invariant sentence(s) can be used as a "signature" to find the decoys, rendering them distinguishable (and clearly, less enticing).

Non-interference: Something that does not hinder, obstructs, or impede.

Introducing decoys to an operational system has the potential to *interfere* with normal operations in multiple ways. Of primary concern is that decoys may pollute authentic data so that their legitimate usage becomes hindered by corruption or as a result of confusion by legitimate users (*i.e.*, they cannot differentiate real from fake). We define non-interference in terms of the likelihood of legitimate users successfully accessing normal documents after decoys are introduced. We use $\text{Access}_{U,m} = 1$ to denote the success of a legitimate user U accessing a normal document m. More formally, for some value ε, the document space M, $\forall m \in M$ we define

$$\Pr[Access_{U,m} = 1] \geq \varepsilon$$

on a system without decoys. Non-interference is then defined for the set of decoys D such that $D \subseteq M$ and $\forall m \in M$ we have

$$\Pr[Access_{U,m} = 1] = \Pr[Access_{U,m} = 1|D]$$

Although we seek to create decoys to ensnare an inside attacker, a legitimate user whose data is the subject of an attacker must still be able to identify their own real documents from the planted decoys. The more enticing or believable a decoy document may be, the more likely it would be to lead the user to confuse it with a legitimate document they were looking for. Our goal is to increase believability, conspicuousness, and enticement while keeping interference low; Ideally a decoy should be completely non-interfering. The challenge is to devise a simple and easy to use scheme for the user to easily differentiate their own documents, and thus a measure of interference is then possible as a by-product.

Differentiable: to mark or show a difference in; constitute a difference that distinguishes; to develop differential characteristics in; to cause differentiation of in the course of development.

It is important that decoys be "obvious" to the *legitimate user* to avoid interference, but "nonobvious" to the insider stealing information. We define this in terms of an inverted believability experiment, in which the adversary is replaced by a legitimate user. We say a decoy is differentiable if the legitimate user always succeeds. Formally, we state this for the document space M with the set of decoys D such that $D \subseteq M$ where

$$\Pr[\mathrm{Exp}_{U,D,M}^{believe} = 1] = 1$$

How might we easily differentiate a decoy for the legitimate user so that we maintain "non-interference" with the user's own actions and legitimate work? The remote thief who ex-filtrates all of a user's files onto a remote hard drive may be perplexed by having hundreds of decoys amidst a few real documents; the thief should not be able to easily differentiate between the two cases. If we store a hundred decoys for each real document, the thief's task is daunting; they would need to test embedded information in the documents to decide what is real and what is not, which should complicate their end goals. For clarity, decoys should be easily *differentiable* to the legitimate user, but not to the attacker without significant effort. Thus, the use of "beacons" or other embedded content in the binary file format of a document, must be judiciously designed and deployed to avoid making decoys trivially differentiable for the attacker.

5 Architecture

The architecture combines host-based user-event monitoring sensors with trap-based decoys and remote network detectors as shown in Figure 1. The combination is designed to make it difficult for insiders to avoid detection with low likelihood of mis-attribution. The architectural components are described in detail in the sections that follow.

5.1 Decoy Document Distributor

One of the core components of the architecture is the Decoy Document Distributor (D^3) System, a web-based service for generating and distributing decoys. D^3 can be used by registered users to generate decoys for download, or as a decoy data source for the host and network components.

The primary goal of a decoy is to detect malfeasance. Since no system is foolproof, D^3 has been built so that multiple overlapping signals may be automatically embedded in decoy documents to increase the likelihood of detecting decoy misuse. Any alert generated by the decoy signals is an indicator that some insider activity has occurred. Since the attacker may have varying levels of sophistication (as discussed in Section 5.3), a combination of techniques is used in decoy documents to increase the likelihood that one will succeed in generating an alert. D^3 generates decoys with several means of detecting their misuse:

- embedded honeytokens, computer login accounts created that provide no access to valuable resources, and that are monitored when (mis)used;
- an embedded "beacon" that alerts a remote server at Columbia, that we call SONAR; and

- an embedded marker, to enable detection by the host-level or network decoy sensor.

These features are explored in the following sections.

Our current deployment of D^3 is tailored for a university environment by both the type of documents and the bait within them, but it can easily be adapted for other deployment environments (*e.g.*, an arbitrary commercial enterprise) . Complete details of D^3 including an evaluation of decoy documents can be found in [2]. The reader is encouraged to visit the Decoy Document Distribution (D^3) website to evaluate our technology developed to date at: `http://www.cs.columbia.edu/ids/RUU/Dcubed`.

5.2 SONAR

SONAR is the alert-management system. Its primary role is to collect alerts triggered by host and network monitors, and individual beacon signals generated by unauthorized opening of decoy documents downloaded by registered users. In response to signals it receives, it emits emails to the registered users associated with the particular decoys. Depending on the type of decoy, some signals are sent directly from a decoy itself (as is the case with beacons), while others require SONAR to poll other resources for information (*i.e.*, credential monitoring). SONAR currently polls a number of servers to monitor credential misuse including university authentication log servers and *mail.google.com* for Gmail account misuse. In the case of Gmail accounts, custom scripts access and parse the bait account pages to gather account activity information.

5.3 Decoys and Network Monitoring

The use of deception, or decoys, plays a valuable role in the protection of systems, networks, and information. Our decoy system builds on the notions of bait information discussed in Section 2 increasing the scope, scale and automation of decoy generation and monitoring.

5.3.1 Perfectly Believable Decoys

In order to create decoys to bait insiders with various levels of knowledge and maximize the deception they induce, one must understand the core properties of a decoy described in Section 4. In this section, we describe our efforts on maximizing the believability of decoys, one of the fundamental properties required to snare an inside attacker; we describe the construction of a “perfect decoy.”

Fig. 1 Architecture

One approach we use in creating decoys relies on a document marking scheme in which all documents contain embedded markings such that decoys are tagged with HMACs [10] (*i.e.*, a keyed cryptographic hash function) and non-decoys are tagged with indistinguishable randomness. Here, the challenge of distinguishing decoys reduces to the problem of distinguishing between pseudo-random and random numbers, a task proven to be computationally infeasible under certain assumptions about the pseudo-random generation process. Hence, we claim these to be examples of perfect decoys and the only attacker capable of distinguishing them is one with the key, perhaps the highly privileged insider. As a prototype perfect decoy implementation, we built a component into D^3 for adding HMAC markers into PDF documents. Markers are added automatically using the iText API, and inserted into the OCProperties section of the document. The OCProperties section was chosen because it can be modified on any PDF without impact on how the document is rendered, and without introduction of visual artifacts. The HMAC value itself is created using a vector of words extracted from the content of the PDF. The HMAC key is kept secret and managed by D^3, where it is also associated with a particular registered host. Since the system depends on all documents being tagged, another component inserts random decoy markers in non-decoy documents, making them indistinguishable from decoys without knowledge of the secret key.

5.3.2 Trap-based Decoys

Our trap-based decoys have an inherent property of being detectable outside of a host, so they do not require host monitoring nor suffer the performance burden characteristic of decoys that require constant internal monitoring. This form of decoy is made up of "bait information" such as online banking logins provided by a collaborating financial institution [2], login accounts for online servers, and web based email accounts. In our current deployment we use Columbia University student accounts and Gmail email accounts as bait, but these can be customized to any set of monitored credentials. The trap-based decoys are managed by the D^3 web service, thereby enabling programmatic access to them from all registered web-enabled clients. The automation of this service enables their distribution and deployment in large volume.

5.3.3 Beacon Decoys

Beacons are implemented to silently contact a centralized server when a document is opened, passing to the server a unique token that was embedded within the document at creation time. The token is used to uniquely identify the decoy document and its association to the IP address of the host accessing the decoy document. Additional data is collected, depending on the particular document type and rendering environment used to view the beacon decoy document. The first proof-of-concept beacons have been implemented in Word and PDF and are deployed through the D^3 website. The Word beacons rely on a stealthily embedded remote image that is rendered when the document is opened. The request for the remote image signals SONAR that the document has been opened. In the case of PDF beacons, the signaling mechanism relies on the execution of JavaScript within the document-rendering application.

The D^3 web service generates many types of beacon decoys including receipts, tax documents, medical reports and other common form-based documents with decoy credentials, realistic names, addresses and logins, information that is familiar to all users. In contrast to the HMAC decoys, the believability of these documents lies in the realism of the content within them.

As noted earlier, the believability of decoys depends on how indistinguishable they are from normal documents. In the case of beacons, the network connection of the beacon may be used as a distinguishing feature. Hence, in their current form the utility of beacon decoys may be limited to ensnaring only the least sophisticated attacker. We are currently investigating environments in which it is possible to embed beacons in all documents, thereby making beacon decoys indistinguishable (modifying the document rendering application is a feasible option). Another potential problem for beacons is that it is possible for the signaling mechanisms to fail or

[2] By agreement, the institution requested that its name be withheld.

be subverted; however, when combined with other mechanisms, we posit their use increases the likelihood of detection.

5.4 Host-based Sensors

One of the key techniques employed by the architecture involves host-level monitoring of user-initiated events. The host sensor serves two functions.

- The sensor is designed to profile a baseline for the normal *search* behavior of a user. Subsequent monitoring for abnormal file search behaviors that exhibit large deviations from this baseline signal a potential insider attack.
- The host sensor also detects when decoy documents containing embedded markers are read, copied, or ex-filtrated. The goal of the host-level decoy sensor is to detect these malicious actions accurately and with negligible performance overhead.

Abnormal user search events that culminate in decoy document access are a cause for concern. A challenge to the user, such as asking one of a number of personalized questions, may establish whether a masquerade attack is occurring. In Section 5.4.1, we present a preliminary evaluation of this sensor.

Our prototype sensor has been built for the Windows XP platform and relies on hooks placed in the Windows ServiceTable. This is a typical approach used by malicious rootkits. However, in contrast to the traditional rootkit objective of remaining undetected, the host-level decoy sensor does not require operational secrecy. Our threat model assumes attackers have knowledge that a system is being monitored, but they must not know the identities of the decoys or the key used by the sensor to differentiate them. Furthermore, the attacker will likely not know the victim user's behavior, information that is not readily stolen such a credential or a key. Given that adversaries may be aware of system monitoring, special care must be taken to prevent the sensor from being subverted or, equally important, to detect if it is subverted. We have ongoing work aimed at preventing and detecting subversion of the sensor. One strategy involves a means to "monitor the monitor" to detect if the host sensor is disabled through the use of tamper-resistant software techniques. One possible solution we are investigating relies on "out-of-the-box" monitoring [6], in which a virtual machine-based architecture is used to conduct host-based monitoring outside of the host from within a virtual machine monitor.

5.4.1 Detecting Anomalous User Search Actions

The sensor collects low-level data from file accesses, windows registry accesses, dynamic library loading, and window access events. This allows the sensor to accurately capture data about specific system and user behavior over time. We posit that one method to check if an insider has infiltrated the system is to model "search"

behavior as a baseline for normal behavior. We conjecture that each user searches their own file system in a manner different from that of a masquerader. It is unlikely that a masquerader will have full knowledge of the victim user's file system and hence may search wider and deeper and in a less targeted manner than would the victim user. Hence, we believe search behavior is a viable indicator for detecting malicious intentions. Specific sections of the windows registry, specific DLLs, and specific programs on the system are involved with system searching applications. For a given time period (10 seconds in our initial experiments), we model all search actions of a user. After a baseline model is computed, the sensor switches to detection mode and alerts if the current search behavior deviates from the user's baseline model. Deviation measurements are made by examining a combination of the volume and velocity of system events in association with other user activities that should add some context to the user search actions, such as the number of processes being created and destroyed. Presently, this sensor component is being integrated in the architecture to function with the host sensor described next that detects decoy document accesses.

The evaluation of our sensor and any insider attack detection approach is made difficult due to the lack of readily available inside attackers or a complete set of realistic data they generate. For this reason, researchers must resort to generating their own data that simulates insider attacks. The Schonlau dataset [15] is the most widely used for academic study. It consists of sequences of 15,000 UNIX commands generated by 50 users with different job roles, but the data does not include command arguments or timestamps. The data has been used for comparative evaluations of different supervised machine learning algorithms. The Schonlau data is not a "true" Masquerade" data set: the data gathered from different users were randomly mixed to simulate a masquerader attack, making the dataset perhaps more suitable for "author identification" studies. An alternative approach to acquire sufficient data for evaluating monitoring and detection techniques is to devise a process to acquire human user data under normal operation as well as simulated attack data where "red team" users are tasked to behave as inside attackers. This type of study is typically subject to Institutional Review Board approvals since human subjects are involved. The process is costly, in time and effort but is sensible and appropriate to protect personally identifiable data of individual volunteer subjects. This was the approach taken by Maloof *et al.* for evaluating ELICIT [11].

We gathered user-event data to compute the baseline normal models, as well as data that simulates masquerade attacks. For the former, we had 34 computer science students at Columbia University install a host sensor on their personal computers. The population of student volunteers assures us the data they generate is derived from sources that have a common "role" in the organization, and hence variations in the user behavior and their data are not attributable to different job functions as is undoubtedly the case with the Schonlau dataset.

The sensor installed by each student monitored all registry-based activity, process creation and destruction, window GUI access, and DLL libraries activity. The data gathered consisted of the process name and ID, the process path, the parent of the process, the type of process action (*e.g.*, type of registry access, process cre-

ation, process destruction, etc.), the process command arguments, action flags (success/failure), and registry activity results. A *timestamp* was also recorded for each action. The collected data was automatically uploaded to a server, after the students had the chance to filter any data that they were not willing to share. The data collected for each user averages about 5 days of normal system use, ranging in the extreme between 1 day and 59 days, and an average of more than 1 million records per user.

To obtain masquerade attack data, we conducted a user study in which 14 students had unlimited access to the same file system for 15 minutes each. None of the users had prior access to this file system, which was designed to look very realistic and to include potentially interesting patent applications, personally identifiable information, as well as account credentials stored in various files. The students were provided a scenario where they were asked to perform a specific task, which consisted of finding any data on the file system that could be used for financial gain.

The features used for modeling were in essence volumetric statistics characterizing search volume and velocity, and describing the overall computer session in terms of number of processes running, particularly the number of editing applications. A one-class Support Vector Machine (ocSVM) model was then trained for each user using those features. The same features were extracted from test data after dividing them into 10-second epochs. The ocSVM models were tested against these features, and a threshold was used to determine whether the user activity during the 10-second epochs was normal or abnormal. If the user activity was performed by the normal user, but was classified as abnormal by the ocSVM model, a false positive is recorded. Our results using the collected data and the modeling approach described above show that, we can detect all masquerader activity with 100% accuracy, with a false positive rate of 0.1%.

Extensive prior work on masquerade attack detection has focused on the Schonlau dataset for evaluation [15]. The data set served as a common gold standard for researchers to conduct comparative evaluations of competing machine learning algorithms. The basic paradigm this work follows is a supervised training methodology where 5000 commands from each user serve as training data for the userÕs normal behavior model. A classifier or model for each user is then tested against hold out data not used in training from the userÕs command dataset but embedded in a random location with another randomly chosen userÕs data. The performance results reported indicate the accuracy of the classifiers learned by a particular machine learning algorithm in identifying Òforeign commandsÓ, those blocks of 100 commands deemed abnormal.

The model we chose to embed in the user search command sensor is different from these prior Òbag of commandÓ oriented models. Our current studies analyze user command events and the rates at which commands are issued using the RUU datasets. Accuracy is estimated with respect to classification errors measured for each 10 second epoch of user events. Furthermore, whereas the Schonalu data consists of Unix commands, the RUU datasets contain user events created in a Windows environment.

In order to compare our results with these prior studies, we need to translate the false positive rates in classifying blocks of 100 commands with the error rate of classifying user commands issued within each standard duration epoch. Unfortunately, the Schonalu dataset is devoid of timestamps and a direct comparison of our modeling technique is not feasible. No one can accurately determine how long it takes each user in the Schonlau data to issue 100 commands. If we assume that it takes 20 seconds to issue one user command on average (a rough estimate from the RUU datasets for certain periods of time), our experiments show a detection rate of 100% can be achieved with a false positive rate of 1.4%. This is a 78% improvement in false positive rate over the best reported classifier in the prior Schonlau work. Indeed, none of the prior work reports a 100% detection rate at any reasonable false positive rate. If we assume it takes on average longer than 20 seconds to issue a user command, the false positive rate drops even further.

The comparison may not be entirely fair since the models and the data are quite different even though the data are generated by human users. The use of temporal statistical features from the RUU data set is crucial in modeling userÕs behavior leading to far more accurate results than blocks of commands. Furthermore, in our work, we focus on user search events, limiting the amount of data analyzed and reducing the complexity of the learning task. Complete details of the volumetric and contextual features we used when modeling user commands and the results achieved are reported in [14] where results of reducing the data in the Schonlau experiments are also described. The RUU datasets, whose size exceeds 8 GBytes, are available to serve as a more up to date and modern gold standard for other researchers to perform similar studies. They can be downloaded at: `http://www1.cs.columbia.edu/ids/RUU/data/`.

5.4.2 Detecting Perfectly Believable Decoys

The second function of the host sensor is to detect malicious activity by monitoring user actions directed at HMAC-embedded decoy documents. Any action directed toward a decoy is suggestive of malicious activity. When a decoy document is accessed by any application or process, the host sensor initiates a verification function. The verification function is responsible for differentiating between decoys and normal documents by computing a decoy HMAC (as described in Section 5.3.1) for the particular document in question and comparing it to the one embedded in the OCProperties section of the document. If there is a match, the document is deemed a decoy and an alert is triggered; otherwise, the document is deemed normal and no action is taken.

The host sensor performs tasks similar to anti-virus programs. In evaluating the performance of the sensor, we use overhead comparisons of anti-virus programs as a benchmark, since the task of comparing an HMAC code is not substantially different from testing for an embedded virus signature. Hence, accuracy performance is not relevant for this particular detector. However, there is a fundamental difference between the task of detecting malware and that of detecting decoy activity. Anti-virus

programs are designed to prevent the execution of and quarantine malicious software whenever any process is initiated. In decoy detection the objective is merely to trigger an alert when a decoy file is loaded into memory. Thus, the decoy detection need not serialize execution; for example, it may be executed asynchronously (and in parallel by running on multiple cores).

We have tested the decoy host sensor on a Windows XP machine. A total of 108 decoy PDF documents generated through D^3 were embedded in the local file system. Markers containing randomness in place of HMACs were embedded in another 2,000 normal PDF files on the local system. Any attempt to load a decoy file in memory was recorded by the sensor including content or metadata modification, as well as any attempt to print, zip, or unzip the file.

The sensor detects the loading of decoy files in memory with 100% accuracy by validating the HMAC value in the PDF files. However, as we discovered during our validation tests, decoy tests can be susceptible to non-negligible false positive rates. The problem encountered in our testing was created by anti-virus scans of the file system! The file accesses of the scanning process that touched a large number of files, resulted in the generation of spurious decoy alerts. Although we are engineering a solution to this particular problem by ignoring automatic anti-virus scans, our test does highlight the challenges faced by such monitoring systems. There are many applications on a system that access files indiscriminately for legitimate reasons. Care must be taken to ensure that only (illicit) human activity triggers alerts. As a future improvement to the sensor, file touches not triggered by user-initiated actions, but rather caused by routine processes, such as anti-virus scanners or backup processes may be filtered. Nevertheless, this demonstrates a fundamental design challenge to architect a security system with potentially interfering competing monitors.

With regard to the resource consumption of the sensor, the components of the sensor used an average 20 KB of memory during our testing, a negligible amount. When performing tests such as the zipping or copying of 50 files, the file access time overhead averaged 1.3 sec on a series of 10 tests, using files with an average size of 33 KB. The additional access time introduced by the sensor is unnoticeable when opening or writing document files. Based on these numbers, we assert that our system has a negligible performance impact to the system and user experience.

6 Concluding Remarks and Future Work

We introduced an architecture that relies on a set of mutually supportive monitoring and auditing techniques. The approach is grounded on the security principle of defense-in-depth and aims to maximize the likelihood of detecting insider malfeasance by employing multiple detection methods. These mechanisms are designed to act synergistically with the goal of making it difficult for an adversary to avoid detection. The architecture combines host-based user-event monitoring sensors with trap-based decoys with the aim of maximizing detection of malicious insider be-

haviors. To aid in deployment of decoys we built D^3, a convenient system to automatically generate and distribute decoy documents. As part of the system, we introduced the concept of the perfectly believable decoy, and developed host-level sensors to demonstrate it with negligible performance overhead. A user search behavior sensor was also presented demonstrating impressive masquerade detection performance far better than previously published results. We posit the integration of all these sensors raises the bar against insider attackers. The risk of detection is likely far greater. Much work remains to be done, particularly on response strategies and system designs that gather and protect evidence to create a demonstrable sense of risk that an insider may be caught and punished for their malicious acts.

The spectrum of techniques we propose covers a broader range of potential attack scenarios than would any of the constituent components in isolation. To date, we have tested and evaluated the individual detectors in isolation, but have not created an integrated end-to-end solution. A fully integrated detection system proposed here cannot be adequately developed, deployed, and formally tested without a fully capable response component, a separate topic beyond the scope of this paper. We must carefully consider the responses to events that are detected by the detectors. For example, should the user be challenged with questions to ascertain whether they are a masquerader, or should a signal alert a system administrator to immediately revoke a credential that is being misused? These questions are context dependent (*e.g.*, determined by an organization's policies) and typically part of product design in a commercial setting. Testing each component detector poses its own challenges due to the lack of generally available insider attack data as discussed in the previous section, which describes the 9 month effort to acquire simulated masquerader data for testing one of the sensors. Acquiring useful "traitor" data to test an integrated system poses challenges we have yet to overcome in a university environment. Even so, we posit that a true controlled study evaluation should be performed when the integrated system responds to insider events it has detected.

Acknowledgments

This material has been published in [2, 3] and has been supported in part by the US Department of Homeland Security under Grant Award Number 60NANB1D0127 with the Institute for Information Infrastructure Protection (I3P), the Army Research Office (ARO) Under Grant Award W911NF-06-1-0151 - 49626-CI, and the National Science Foundation (NSF) under Grant CNS-07-14647. 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, ARO, NSF, or Dartmouth College.

References

1. Bell, J., Whaley, B.: Cheating and Deception. Transaction Publishers, New Brunswick, NJ (1982)
2. Bowen, B.M., Hershkop, S., Keromytis, A.D., Stolfo, S.J.: Baiting inside attackers using decoy documents. In: In Proceedings of the 5^{th} International ICST Conference on Security and Privacy in Communication Networks (SecureComm 2009) (2009)
3. Bowen, B.M., Salem, M.B., Hershkop, S., Keromytis, A.D., Stolfo, S.J.: Designing host and network sensors to mitigate the insider threat. In IEEE Security & Privacy Magazine **7**(6), 22–29 (2009)
4. (2009). URL `http://www.cs.cornell.edu/bigreddata/cayuga/`
5. Ilett, D.: Trojan attacks microsoft's anti-spyware (2005)
6. Jiang, X., Wang, X.: "Out-of-the-Box" monitoring of vm-based high-interaction honeypots. In: Proc. of the 10^{th} International Symposium on Recent Advances in Intrusion Detection (RAID), pp. 198–218. Cambridge, MA, USA (2007)
7. Katz, J., Lindell, Y.: Introduction to Modern Cryptography: Principles and Protocols. Chapman & Hall/Crc Cryptography and Network Security Series (2007)
8. Lee, W., Fan, W., Miller, M., Stolfo, S.J., Zadok: Toward cost-sensitive modeling for intrusion detection and response. In: Workshop on Intrusion Detection and Prevention, 7^{th} ACM Conference on Computer Security, November 2000 (2000)
9. Li, W., Stolfo, S.J., Stavrou, A., Androulaki, E., Keromytis, A.D.: A study of malcode-bearing documents. In: Proceedings of the 4^th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment(DIMVA), pp. 231–250 (2007)
10. Maloof, M.A., Stephens, G.D.: Keying hash functions for message authentication. In: Proceedings of the 16^{th} Annual International Cryptology Conference on Advances in Cryptology, pp. 1–15 (1996)
11. Maloof, M.A., Stephens, G.D.: Elicit: A system for detecting insiders who violate need-to-know'. In: Recent Advances in Intrusion Detection (RAID 2007), pp. 146–166 (2007)
12. (2009). URL `http://www.oakleynetworks.com/products/sureview.php`
13. Richardson, R.: Csi computer crime and security survey. Technical report, CERT (2008)
14. Salem, M.B., Stolfo, S.J.: Masquerade attack detection using a search-behavior modeling approach. Technical report, Columbia University (2009)
15. Masquerading user data (2009). URL `http://www.schonlau.net/intrusion.html`
16. Spitzner, L.: Honeypots: Catching the insider threat. In: Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC), pp. 170–179 (2003)
17. Spitzner, L.: Honeytokens: The other honeypot. Technical report, SecurityFocus (2003)
18. Stoll, C.: The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage. Pocket Books, New York (1990)
19. Symantec: Trends for july - december '07. White paper (2008)
20. (2009). URL `http://www.verdasys.com`
21. Webb, S., Caverlee, J., Pu, C.: Social honeypots: Making friends with a spammer near you. In: In Proceedings of the Fifth Conference on Email and Anti-Spam (CEAS 2008) (2008)
22. Yuill, J., Denning, D., Feer, F.: Using deception to hide things from hackers : Processes, principles, and techniques. Journal of Information Warfare **5**(3), 26–40 (2006)
23. Yuill, J., Zappe, M., Denning, D., Feer, F.: Honeyfiles: Deceptive files for intrusion detection. In: Proceedings of the 5th Annual IEEE SMC Information Assurance Workshop (IAW), pp. 116–122 (2004)

Insider Threat Specification as a Threat Mitigation Technique

George Magklaras and Steven Furnell

Abstract Insider threats come in many facets and nuances. This results in two major problems: mining big amounts of data for evidence of an insider attack, and keeping track of different aspects of threats is very cumbersome. To enable techniques that support detection of insider threats as early as possible, one needs mechanisms to automate significant parts of the detection process, and that allow to specify what is meant by insider threat. This chapter describes the Insider Threat Prediction Specification Language (ITPSL), a research effort to address the description of threat factors as a mechanism to mitigate insider threats.

1 Introduction

Insider threats come in many facets and nuances. The goal of automated detection techniques that support detection of insider threats as early as possible is to enable Information Security Management tools to specify what is supposed to be an insider threat, and to use these mechanisms for detecting realised threats in the observed behavior.

This goal results in two major problems: the amount of data to be mined is huge, if one wants to achieve a reasonable success rate, and keeping track of different aspects of threats is very cumbersome.

George Magklaras
Centre for Security, Communications and Network Research, University of Plymouth, Plymouth, United Kingdom, and
The Biotechnology Center of Oslo University of Oslo, Norway, e-mail: georgios@biotek.uio.no

Steven Furnell
Centre for Security, Communications and Network Research, University of Plymouth, Plymouth, United Kingdom, e-mail: S.Furnell@plymouth.ac.uk

C. W. Probst et al. (eds.), *Insider Threats in Cyber Security,*

DOI 10.1007/978-1-4419-7133-3_10,

Based on developments in the direction of specification languages and domain-specific languages, we have developed the Insider Threat Prediction Specification Language (ITPSL), a research effort to address the description of threat factors as a mechanism to mitigate insider threats. Once we are able to express what we consider as a threat, we can hope to use automatic tools for mining the observed data for occurrences of what is described.

Before describing ITPSL itself we provide some necessary definitions about the insider misuse problem in the rest of this section. It is followed by the description of the background for the development of our language in Section 2, where we start with a discussion of two intrusion specification language paradigms that influenced the development of ITPSL. In Section 3 we discuss taxonomic and threat modeling research and development efforts designed to address insider threats, with emphasis on abstracting the domain of insider misuse and shaping the threat metrics the language can express. Section 4 explains the problems ITPSL is trying to solve and its design criteria, and Section 4.1 discusses the programming paradigm that could facilitate the ITPSL construction. Finally, Section 5 concludes this chapter.

1.1 The Insider Threat Problem

The problem of insider IT misuse (the term 'misuse detection' or 'misuse' is also used in the literature) is a serious threat for the health of IT infrastructures. A threat in an IT infrastructure context is "a set of circumstances that has the potential to cause loss or harm" [30]. As a result, in legitimate user context, these circumstances might involve intentional IT misuse activities such as targeted information theft, introducing or accessing inappropriate material, and accidental misuse, *e.g.*, unintentional information leak. In addition, there is also potential for flaws in the design and implementation of the computer system, which could render it susceptible to insider misuse.

Numerous people have tried to define the term "insider" in the context of Information Security. This is because there are many possible sub-contexts that are applicable to shedding light on different aspects of what an insider is and what she can do. For instance, an aspect of insiders relates to what they are allowed and not allowed to do in an organizational context. This is often dictated by the organization's IT usage policy, "a set of laws, rules, practices, norms and fashions that regulate how an organisation manages, protects, and distributes the sensitive information and that regulates how an organisation protects system services" [7]. Insiders that do not follow the rules of the IT policy are formally considered as misusers.

Other definitions focus more on the attributes of an insider, from an organizational trust point of view [5]:"An insider is a person that has been legitimately empowered with the right to access, represent, or decide about one or more assets of the organization's structure". This definition has a wide perspective and emphasizes a key aspect of an insider: that of trust. Trust is a property that goes beyond an IT system oriented view (system credentials, actions, indications). Whilst people who

constitute direct threats might not have access to IT access credentials, they still can decide on policies, equipment procurement and other issues that can affect the well being of an IT infrastructure. A good example is an IT director that spends millions on a state of the art security system but does not bother to emphasize or make policies that dictate the flow of information inside the organization (employ that bypasses the system with a simple USB key, intentionally or accidentally).

However, trust has an impact on IT level credentials. A narrower but IT system specific definition can also be useful, in order to focus on insider actions that can be detected by system methods. Hence, an insider is a person that has been legitimately given the capability of accessing one or many components of an IT infrastructure (hardware, software and data) enjoying effortless login by interacting with one or more authentication mechanisms. The word 'legitimately' differentiates the user from an external cracker that masquerades as the user by means of bypassing the authentication mechanisms. The implication of 'effortless' is that an insider does not need to consume time and effort to gain access to a system resource. This also means that they enjoy trust, a vital property of all insiders.

The reader can consult references such as [32, 6, 10] for a detailed qualitative and quantitative review of insider misuse cases.

2 Background

The main function of ITPSL concerns insider threat specification. Threat specification is not a new concept in the information security world. Techniques to describe threats exist for an entire range of information security products, from anti-virus software to several intrusion detection/prevention systems, where specified rules are used to describe a particular range of threats. However, this section focuses on generic threat specification. Most products might focus on specific types of threats (anti-virus products relate to malware detection, IDS products might focus on network threats, etc).

When it comes to generic intrusion specification languages, we have two notable examples. The Common Intrusion Specification Language (CISL) [13] and Panoptis [40]. The next paragraphs are going to describe these two languages and discuss their significance for ITPSL.

2.1 The Common Intrusion Specification Language

The Common Intrusion Specification Language (CISL) consists of a semantic framework to unambiguously describe intrusive activities together with proposed data structures that store the event information and can form standardized messages exchanged by various Intrusion Detection System (IDS) components. The semantic representation of intrusive activities is achieved by the formation of an S-

Expression. This is a recursive grouping of tags and data, delimited by parentheses. The tags provide semantic clues to the interpretation of the S-Expression and the data might represent system entities or attributes. For this reason, the tags are also called Semantic Identifiers (SIDs).

The best of way of illustrating how CISL works is by considering an example. The statement (Hostname 'frigg.uio.no') is a simple S-Expression. It groups two terms, without semantically binding them. One can guess that it refers to a computer system with the FQDN name 'frigg.uio.no', but the true meaning of the statement is still vague. In fact, the full semantic meaning of S-Expressions becomes apparent when one forms more complex S-Expressions, by means of combining several SIDs into a sentence.

Figure 1 illustrates a suitably crafted CISL intrusion specification that could be translated in the following plain English translation:

> *"On the 24th of February 2005, three actions took place in sequence in the host 'frigg.uio.no'. First, someone logged into the account named 'tom' (real name 'Tom Attacker') from a host with FQDN 'outside.firewall.com'. Then, about a half-minute later, this same person deleted the file '/etc/passwd' of the host. Finally, about four-and-a-half minutes later, a user attempted but failed to log in to the account 'ksimpson' at 'frigg.uio.no'. The attempted login was initiated by a user at 'hostb.uib.no'."*

The particular CISL sentence describes a malicious attack that erases an important system file of a UNIX system and consists of three multi-SID S-Expressions. In general, a sentence can be formed by one or more S-Expressions nested at different levels.

Verb SID's are joined together in a sentence by conjunction SIDs. In the previous example of Figure 1, 'And' is the conjunction SID that holds together the three SIDs that form the sentence. In addition, a CISL sentence might employ role, adverb, attribute, referent and atom SID types. Role SIDs indicate what part an entity plays in a sentence (such as 'Initiator'). Adverb SIDs provide the space and time context of a verb SID. Attribute SIDs indicate special properties or relations amongst the sentence entities, whereas atom SIDs specialise in defining values that are bound to certain event instances (for instance 'Username'). Lastly, referent SIDs allow the linking of two or more parts of a sentence ('Refer to' and 'Refer as'). There are additional SID types, but the aforementioned ones are the most commonly employed.

One can clearly observe a structural hierarchy for forming complex sentences that also contributes to the semantic meaning. This semantic structure is inspired by the syntax of natural languages. A verb is always at the heart of every sentence and is followed by a sequence of one or more qualifiers that describe the various entities that play parts in the sentence, or qualify the verb itself.

CISL [13] is not only about semantic rules. Its authors were concerned with the encapsulation of the structured semantic information into the 'Generalised Intrusion Detection Object' (GIDO), data structures that hold the encoded event information. The purpose of encoding the information in a standard way is to make the process of exchanging the information amongst various IDS components easy.

Unfortunately, despite the well-conceived interoperability target, the CISL GIDO encoding process introduced many problems. Doyle [12] has criticized many of the

```
(And
        (OpenApplicationSession
            (When (Time 14:57:36 24 Feb 2005) )
            (Initiator (HostName 'outside.firewall.com') )
            (Account
                (UserName 'tom')
                (RealName 'Tom Attacker')
                (HostName 'frigg.uio.no')
                (ReferAs 0x12345678)
            )
            (Receiver (StandardTCPPort 22) )
        )
        (Delete
            (World Unix)
            (When (Time 14:58:12 24 Feb 2005) )
            (Initiator (ReferTo 0x12345678) )
            (FileSource
                (HostName 'frigg.uio.no')
                (FullFileName '/etc/passwd')
            )
        )
        (OpenApplicationSession
            (World Unix)
            (Outcome
                (CIDFReturnCode failed)
                (Comment '/etc/passwd missing')
            )
            (When (Time 15:02:48 24 Feb 2005) )
            (Initiator (HostName 'hostb.uib.no') )
            (Account
                (UserName 'ksimpson')
                (RealName 'Karen Simpson')
                (HostName 'frigg.uio.no')
            )
            (Receiver (StandardTCPPort 22) )
        )
    )
```

Fig. 1 CISL sentence syntax example.

aspects of the CISL GIDO structure. Although the purpose of the document was to evaluate the fitness of CISL for use in the DARPA Cyber Command and Control (CC2) initiative, the paper identifies serious inadequacies that concern the CISL time resolution data representation facilities, as well as data throughput limitations caused by the fixed size of the GIDO data structure. Finally, Doyle comments on the lack of support for the next generation Internet Protocol (Version 6). Whilst these points are fair, they could easily be corrected by making the necessary changes to the relevant data types and overcome the perceived obstacles. In fact, Section 7 of the CISL standard [13] contains specific guidelines that explain how to add infor-

mation to a GIDO, to clarify or correct its contents. This suggests that the encoding principles are certainly extensible.

A more serious aspect of Doyle's critique [12] refers to the semantic structure of the CISL language. In particular, his criticism that CISL has "no facilities for representing trends or other complex behavioral patterns; ill-specified, inexpressive, and essentially meaningless facilities for representing decision-theoretic information about probabilities and utilities" indicates that the language would be a bad choice for describing threat prediction related information. The basic reasoning behind this critique is that CISL is too report-orientated and threat mitigation requires a different level of information, not just mere report structures of what is happening on one or more systems. These indeed represent more serious limitations that would require a more radical re-design of the CISL.

In addition to Doyle's criticisms, from a threat specification perspective, we note the following omissions/weaknesses in CISL:

1. Inability to express variability in intrusion events: For example, all the necessary time patterns to specify recurring events of significance: The 'When', 'Time' and others SIDs can bind an event to an accurate time and date location. However, this is of little value to a threat specification as the accurate time of an intrusion event is rarely known. An SID operant such as 'afternoon-hours', 'evening-hours' would be more functional. This is true for other type of SIDs such as 'FileSource', network SIDs, etc. The expressions clearly lack the necessary polymorphism required to describe a range of possible events.
2. The nesting of S-Expressions does not facilitate logical operands/operators, in order to describe alternative events. This affects the overall polymorphic description at event level.
3. General lack of a mechanism to express confidence of a particular metric: Decision theoretic information is a desired feature of threat specification. The process of specifying a threat might include the description of various events. Not all of them have the same level of reliability and as such, a language that omits a mechanism of expressing a confidence in a particular event is a serious issue. This omission also hinters the ability to build up user profiling information.

Nevertheless, Amoroso [3] characterizes CISL [13] and its associative CIDF framework [10] as a "good piece of computer science", despite the fact that it has not managed to infiltrate the IDS/IPS vendor market as a product interoperability platform. CISL is significant for the development of ITPSL for the following reasons:

- It is the first language framework for generic intrusion specification with system interoperability as its design goal, attempting to bridge the gap between language semantics and operating system/IDS product implementation details. This is a desirable feature because a good threat specification mechanism should focus on the threat itself and less on platform specific issues.
- It introduces the S-expression as a way to group the SIDs with the actual data in a hierarchical semantic notation which can nest expressions. Despite the previously discussed weaknesses of its proposed semantic identifiers, the suggested

```
HZ = 100                # "Floating point" value divisor
bigend = FALSE          # Set according to machine
map = TRUE              # Set to TRUE to map uid/tty numbers to names
EPSILON = 150           # New maxima difference threshold (%)
report = TRUE           # Set to TRUE to report new/updated entries
unlink = FALSE          # Set to TRUE to start fresh

# Reporting procedure
output = "| /usr/bin/tee /dev/console | /bin/mail root"

# Databases and parameters to check
dbcheck(tty, minbmin, maxbmin, maxio, maxcount)    # Terminals
dbcheck(comm, ALL)                                 # Commands
dbcheck(uid, ALL)                                  # Users
dbcheck(uidtty, maxcount)                          # Users on a terminal
dbcheck(uidcomm, minbmin, maxbmin, maxutime,       # command
        maxstime, maxmem, maxrw, maxcount, maxasu)

# Map users and terminals into groups

usermap(caduser, john, marry, jill)
usermap(admin, root, bin, uucp, mail, news)
```

Fig. 2 A configuration file sample showing the DSL syntax of Panoptis.

combination increases the clarity of the expression and the S-expression nesting capability increases the specificity of the statement in a consistent manner (the more S-expressions nested together in a the more specific the conditions of the match).

2.2 Panoptis

Panoptis [40] is another interesting and more recent intrusion specification language paradigm. The language sits on top of an anomaly detection system which parses standardized UNIX audit process logs. After establishing a user profile based on a number of different criteria, the audit logs are parsed and then checked against the profiling data. A sample of the entities and quantitative criteria that the Panoptis system checks against includes data on discrete entities such as uses and terminals, commands executed, process accounting data, etc.

In essence, Panoptis is an anomaly detection system envisaged to detect a number of attacks such as data leakage, wiretapping and user masquerading amongst others. The semantics are restricted to configuration file options such as the one illustrated by Figure 2. Declarations of the type *variable=value* and a *keyword(entity, value(s))* combination make up the syntactical convention.

For instance, the statement *dbcheck(tty, minbmin, maxbmin, maxio, maxcount)* will check the UNIX terminal entity activity against the normal minimum and maximum startup time, as well as the maximum character input/output and the maximum number of times a given record has appeared in the database. If any of these figures exceeds the preset epsilon threshold normal value by 150% (*EPSILON=150* declaration), the observation will be flagged as an intrusion. Note that these checks will be performed against the records of certain users as defined by the *usermap* statements (*caduser, john, marry, jill* as user group 1 and *admin, root, bin, uucp, mail, news* as user group 2).

The simplistic semantics of Panoptis suffer from many of the previously discussed drawbacks of CISL. Development on the 'panoptis' system has been discontinued and thus, it is not fair to really judge the effort on the basis of the presented system. The panoptis authors were more interested to present a paradigm whose scope was to parse system audit logs and not a full intrusion specification language.

However, the Panoptis approach is an important paradigm for ITPSL for two reasons:

- It is one of the first specification language approaches that target insider misuse incidents. The authors claim that under certain conditions, panoptis could "detect an employee transferring inordinately large amounts of datato a computer outside the organisation even if that employee had proper system authorisations to perform" [40].
- It is one of the first systems that employs a Domain Specific Language approach, in order to design the intrusion specification semantics and capture precisely the domain's semantics. Section 4.1 will examine the Domain Specific Language more closely.
- It makes use of simple data management techniques in terms of having access to structured data input (reading from UNIX process audit logs) and arranging the anomaly detection threshold in a simple non-relational database. This indicates the need for properly storing and readily accessing intrusion information, an important requirement of a threat specification tool in itself.

3 Insider Misuse Taxonomies and Threat Models

Apart from the process of designing the semantics of the specification language, there are other steps that concern the language designer. In fact, we have earlier [23] proposed a methodology for deriving a Domain Specific Language for insider misuse detection and prediction that includes three important components:

- the abstraction of the domain, which involves the removal of all the unnecessary details of the environment;
- the systematic categorisation of the necessary (abstracted) details into language semantics;
- the process of engineering the developed semantics into software.

The domain abstraction is a critical part of the overall language design process and raises the question of which entities and data are relevant to the insider threat domain. The CISL and Panoptis paradigms discussed above propose certain types of data to monitor, without giving a concrete explanation on why these data were chosen and how they can aid the threat detection/prediction process. Section 1.1 discussed some insider misuse case studies and surveys and concluded that whilst generic trends can be spotted, this is not enough information to have a concrete picture of the problem. In order to select with confidence a range of insider misuse threat descriptors, a more systemic view of the problem is needed.

Information security taxonomies and threat models provide the answer to these questions. Taxonomies are efforts to classify information. Threat Modeling attempts to make use of the systemic knowledge of the taxonomies and estimate threat levels and/or simulate threat scenarios to help insider misuse researchers understand better important concepts and ultimately estimate threat levels. A model is an "abstraction of the system being studied rather than an alternative representation of that system" [38]. This abstracted representation of the system should closely resemble its real-world behavior. The process of abstracting a real-world situation implies that not all information about its attributes and functions is transferred into the model. Only those attributes and functions that are relevant for the study of certain aspects of the entities involved are included. Thus, insider threat model designers need to consider carefully which attributes and behavioral characteristics of a legitimate user are important to a threat estimation process.

As with intrusion specification languages (Section 2), intrusion specification taxonomies are not a new idea in the information security field. An overview of intrusion specification taxonomies is provided by Furnell *et al.* [15]. Amongst these taxonomies, one that specifically addresses insider IT misuse incidents is given by Tuglular [42]. Tuglular is one of the first to suggest a 'target-type of threat' association as a way to prevent insider misuse. The target is an 'asset' and the rule is called a 'strategy' in the terminology he proposes. The suggestion forms the basis for a methodology to predict insider misuse threats. If one can associate successfully certain actions to threats then it establishes the first step towards systematizing insider IT threat prediction.

Most research efforts in the field of intrusion taxonomy classification are still at an early stage. The Tuglular taxonomy, and others mentioned in [15], are useful for the systematic study of intrusions, but they offer little help to a process designed to automatically detect intrusive activities. This is because the classification criteria employed by these taxonomies cannot be qualified or quantified very easily by an Intrusion Detection System with the level of information they exhibit. Moreover, none of these taxonomies is tailored for the process of estimating the likelihood of Insider Threat.

The best way of enhancing the expressiveness of an intrusion taxonomy scheme for insider misuse activities is to focus on the human actions and how their consequences impact the elements of the IT infrastructure that are being targeted. The idea is that it is easier to detect which particular element is affected by a potentially intrusive action, rather than focusing on the task of sensing the motives for initial-

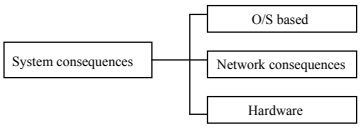

Fig. 3 Top hierarchy level of an insider misuse taxonomy.

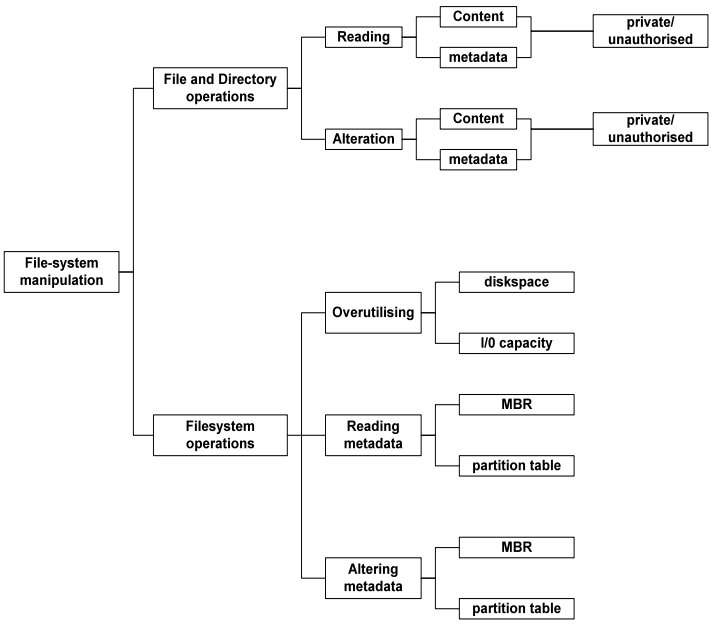

Fig. 4 File-system manipulation O/S consequences.

izing an attack or focusing on other non-system detectable factors of the insider misuse domain.

As discussed earlier, the perplexing variable nature of insider IT misuse is a fact. What is considered as misuse by a well-defined IT usage policy is not the same across different organizations. The freedom of the security architect to choose what can be considered as an Insider IT misuse threat indicator and even decide on the confidence of each indicator is important. Most taxonomies enforce a rigid framework for classifying phenomena with clear borders of distinction that offer little space for subjective or varying interpretation of facts. This schema does not fit the case of Insider IT misuse prediction.

Figure 3 below displays the top level of the taxonomy structure indicating the three primary, non-mutually exclusive levels that address these consequences.

The Operating System (O/S) based consequences are branched down to two sub-levels of file-system and memory manipulation, illustrated by Figure 4 and Figure 5 respectively. A justification for this is that a large number of security faults [41] involve filesystem and memory management issues, and indeed the core modules

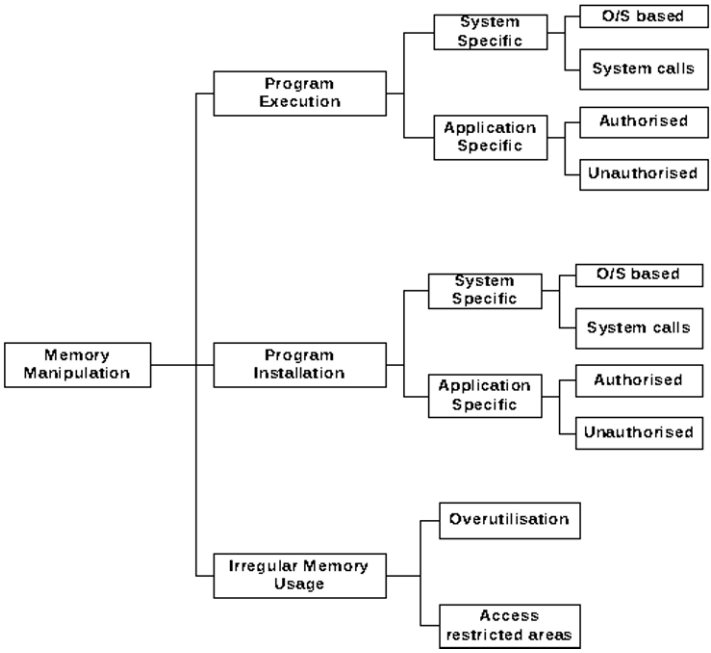

Fig. 5 Memory Manipulation O/S Consequences.

of UNIX [4] and Windows-based [34] operating systems provide (amongst others) specific support for the related functions. Hence, it is safe to assume that these two kernel functional attributes can be used as a strong criterion for further classifying legitimate user activities.

At File/Directory level, a misuser may attempt to read or alter (write/create) certain files. These files might contain sensitive or unauthorised information (information theft or fraudulent modification of vital information). A knowledgeable insider might also attempt to read or modify file information that is not directly related to its content. Bach [4] and Richter [34] emphasize that most Operating Systems allow a file to contain additional information such as access/creation/modification times as well as information that relates the file to its owner and permits access to it under certain conditions. Although the mechanisms that implement these file attributes are different amongst Operating Systems, they are collectively known as file metadata and they are vital mechanisms to secure the privacy, availability and integrity of the file contents. Consequently, they are good candidates for exploitation by a legitimate user who is about to perform a deliberate or accidental misuse act.

The points mentioned in the previous paragraph are also valid for 'filesystem' related data. Every Operating System organizes its files and directories by means of a specific set of rules that define how a file (contents and metadata) are about to be stored on the physical medium. The Operating System sub-modules that handle these issues are known as filesystems. Attempts to read or alter the physical

medium's Master Boot Record (MBR), intentional or accidental modification of partition table data are some of the most notable auditable actions that could point to legitimate user misuse acts. Robert Hanssen's attempt to hide information in modified floppy disks, a case [32] discussed in Section 1.1, is a classic reminder of this kind of activity.

In addition to filesystem content and metadata modification, a survey of insider misuse [17] showed that excessive disk space consumption is perceived as a problem for many of the respondents. Under certain conditions that depend on the configuration of the IT infrastructure, a legitimate user might produce a deliberate or accidental Denial of Service attack (DoS).

Memory inspection is the best way to see if a legitimate user attempts to run or even install a suspicious program. Indeed, it is one of the core techniques used in the detection of overtly malicious code, such as viruses and Trojan horse programs. The usage of unauthorised programs is a serious issue that can also create a way for accidental misuse by introducing a number of system vulnerabilities [16]. The execution or installation of these programs could be intercepted by either recognising a program's footprint in memory or by intercepting a well-known series of system calls produced by various suspicious programs. For example, the fact that a non-advanced user is trying to compile an advanced vulnerability scanning tool is an event that should be noticed and serve as a good indicator of potential misuse activities that are about to follow.

In addition, attempts to consume large memory portions of an operational system that are related to a legitimate user account can serve as good indicators of (intentional or accidental) insider misuse at Operating System level. One might argue that the 'irregular memory usage' sub-categories should really belong under the 'Program execution' hierarchy of events. However, it is possible that someone will produce a quick and easy Denial of Service attack on a running system by forcing the host to commit large portions of system memory to a process, as demonstrated in various case studies described by Moore *et al.* [26]. Moreover, a large category of security faults can be achieved by means of accessing normally restricted memory areas, creating what is commonly known as a "buffer overflow" attack [14]. As a result of these issues, it was felt that a separate sub-category hierarchy should exist to describe these events (Figure 5).

The filesystem and memory manipulation consequences conclude the O/S consequence category of the proposed taxonomy (Figure 3). The next category, "network consequences", represents another distinct set of factors that could be taken into consideration in order to classify insider misuse threat indicators. Figure 6 illustrates the network-related consequences of acts that could be used as legitimate user threat indicators.

Network packets that are associated with certain legitimate users and indicate the usage of a variety of network protocols and applications that might introduce certain vulnerabilities are also distinct ways of accidental or intentional IT misuse. For example, it could be said that a user that employs the TELNET protocol [31] to login to a multi-user system is more likely to have her account compromised than a user who logins via the Secure Shell (SSH) application [44] due to the fact that the

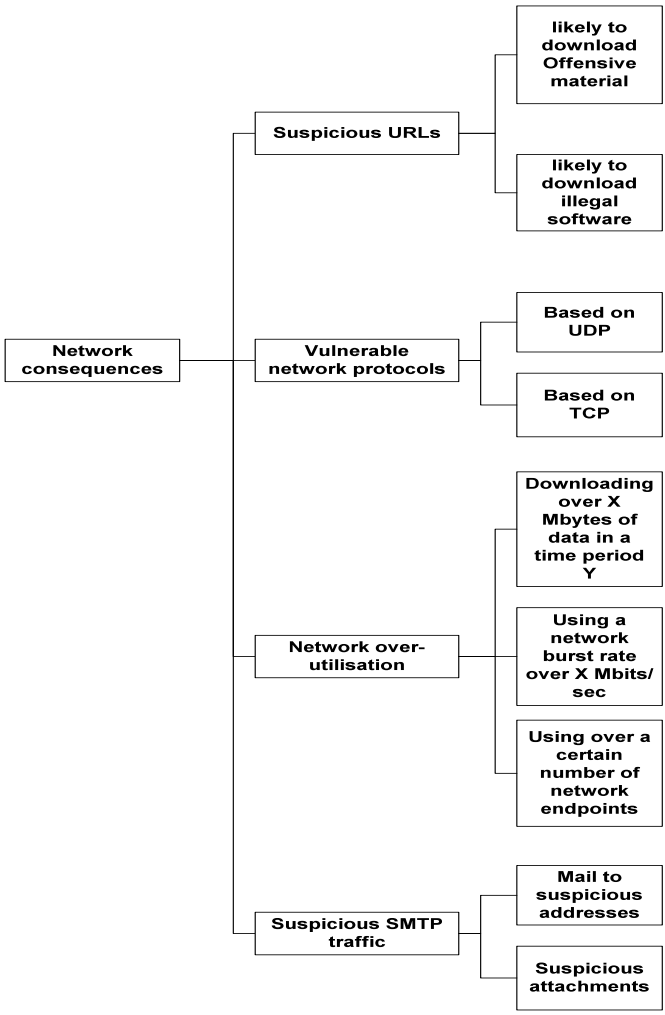

Fig. 6 Network consequences of the insider IT misuse prediction taxonomy.

earlier application transmits the user password in clear-text form across the network, whereas the latter one encrypts it.

Someone might also like to differentiate between TCP and UDP based applications/protocols. From a potential threat point of view, UDP services are less secure than TCP based ones. For example, Ziegler [45] discusses in detail how UDP's lack of flow control and state mechanisms can create various data security problems. Consequently, the distinction between the usage of UDP and TCP services can serve as a potential insider misuse threat indicator, on the basis that UDP services are more likely to be accidentally (or intentionally) abused by a legitimate user.

Although the 'Filesystem Manipulation' subcategory of the taxonomy indicates ways with which disk storage capacity can be misused, the results of over-utilisation can also affect network capacity. For instance, a legitimate user could start downloading massive quantities of data, exceeding the network bandwidth cost budget of a business (Downloading over X Mbytes of data in a period Y). The X and Y number limits can be selected by the network administrator according to the company budget requirements.

In addition, a legitimate user might also cause network congestion by exceeding the data network's 'burst' or throughput capacity or exhausting the number of available network endpoints, as described by Sharda [36]. Bandwidth hungry applications, such as video streaming players, and multiple data transfers can cause congestion that can severely impact the performance of a data network or affect the Quality of Service (QoS) of certain applications that require sustained data network throughput.

Finally, incoming or outgoing SMTP headers or attachments might indicate activity related to e-mail misuse that can certainly be traced in network or host level. Outgoing e-mails that contain a set of particular files as attachments (*e.g.*, password database files, other sensitive material) and have unusual destination addresses (*e.g.*, unknown Hotmail accounts, a large number of recipients) should serve not necessarily as intrusion indicators but as insider threat estimators. The plethora of malicious code efforts and phishing techniques may have an external origin, but the threat is realized by the actions of unsuspecting legitimate users. In addition, proprietary information theft could also be realized by means of emailing sensitive material to non-authorized external entities.

The last system consequences category ("hardware") plays an important role in preventing a number of computer system threats. Insiders can often access the physical hardware of the machine very easily. Thus, removal or addition of hardware components, as well as modifications of their default configuration are some of the events that may act as important indicators of insider misuse prediction in a computer system.

However, in order to make use of these threat indicators, we need a way to quantify them. This paves the way for the discussion of various Insider Threat Models presented in the following paragraphs.

The first important step of deriving an Insider Threat Prediction Model is to decide which attributes and behavioural (functional) characteristics of a legitimate user are important to the Threat Estimation Process. This will produce a set of Insider Threat Qualification Attributes (ITQAs).

The next step in the process of establishing the model is to describe how the ITQAs can be quantified, in order to estimate the level of insider threat per individual user. This will involve the establishment of a suitable mathematical function, which will take as input a number of ITQAs and will associate them with a certain level of threat. We shall call this function the Estimated Potential Threat function, which quantifies the ITQAs. At this point, the overall target of our ideal model will be achieved: the establishment of a mechanism that will map ITQAs to certain threat levels.

The development of insider threat models is a relatively new idea. Wood [43] provides an excellent basis for qualifying a set of metrics to mitigate insider threat. Most of these criteria are in line with the conclusions issues discussed as part of the insider misuse taxonomy discussed in the previous section.

In particular, Wood suggests that a malicious insider can be qualified in terms of distinct attributes:

- Access: The insider has unlimited access to some part or all parts of the IT infrastructure and the ability to physically access the equipment hardware. Consequently, the insider can initiate an attack without triggering traditional system security defences.
- Knowledge: The legitimate user is familiar with some or all the internal workings of the target systems or has the ability to obtain that knowledge without arousing suspicion.
- Privileges: The malicious insider should not have problems obtaining the privileges required to mount an infrastructure attack.
- Skills: The knowledgeable insider will always have the skills to mount an attack that is usually limited to systems that he/she is very familiar with. The model assumes that a given adversary is unlikely to attack unfamiliar targets.
- Tactics: This attribute refers to the methods used to launch the malicious attack. They are dependent on the goal of the attack and might include a variety of scenarios such as plant-hit-and-run, attack-and-eventually run, attack-until-caught as well as passive information extraction acts.
- Motivation: Insiders might launch the attack for profit or sabotaging the target organisation. Some of them might mount an attack for personal reasons such as taking revenge against the enterprise or even satisfy their plans to invoke some policy change inside an organisation.
- Process: The model assumes that a legitimate user follows a basic predictable process to mount an attack that consists of distinct stages. First the malicious adversary will become motivated to mount the attack. The next logical stages involve the identification of the target, the planning of the attack and finally the act of mounting the attack itself.

All of the previously mentioned attributes emphasize important aspects of the insider misuse problem. However, Wood's criteria do not necessarily represent a clear picture for the establishment of an insider threat prediction model. Not all stages of an insider attack can be safely predicted. Some of the previously mentioned attributes are difficult to qualify by an Intrusion Detection System. The 'motivation' adversary attribute is one of them.

It is very difficult to establish a set of sensors that could reliably deduce when an individual becomes motivated to misuse a system. For instance, let us suppose that IDS sensors record that a commercially important file is transferred from a disk to an external storage medium in the early morning hours. The fact that this particular file transfer took place could be related to a malicious act or an innocent file backup process performed by the system administrator as part of a system recovery process. It is important to maintain a record of these types of events, but their existence does

not necessarily indicate an insider misuse event in progress. The plethora of the potential origins of such an event would increase the amount of information to be evaluated. Consequently, the complexity of the algorithms to capture and evaluate this type of information would deem this attribute's exploitation impractical.

If someone observes the different stages of the 'process' insider-modelling attribute, it becomes clear that the closer we get to the actual attack itself, the stronger the indicators of insider threat. Although detecting motivation might be tricky, with a carefully chosen quantification scheme of ITQAs, someone could sense an adversary during the target identification and attack planning stages. This strategy goes along the line of thinking of our proposed misused taxonomy being based on system-level factors.

In addition, other attributes seem to be so closely related that might be redundant. For instance, it would be more logical to combine the attributes of 'access' and 'privileges' into one 'insider access rights'. The issue of obtaining a privilege to mount an attack should include logical and physical means of interacting with the systems. The same could be said for the attributes of 'knowledge' and 'skills', because the ways in which a legitimate user gets to know a system and what can be inferred from the insider's system knowledge are issues that are closely interrelated.

A more recent research effort by Schultz [35] presents a preliminary framework for understanding and predicting insider attacks by providing a combination of behavioural and system usage ITQA metrics. The paper mentions the detection of system usage patterns that may act as "signatures" of a legitimate user or certain indicators of an attack preparation ("deliberate markers" and "preparatory behaviour"). Legitimate users might also make noticeable mistakes in the process of misusing a system (meaningful errors). Finally, "correlated usage patterns" refers to sequences of actions that might not be detected in individual systems but they could certainly indicate misuse when considered against multiple systems.

Schultz also suggests that certain aspects of a legitimate user's personality could serve as threat indicators. In particular, on-line (e-mail, IRC or other forms of computerised human-to-human communication) verbal behaviour with signs of aggression, dominance towards particular people might serve as a good prognosis factor of certain attacks ("verbal behaviour"). Furthermore, based on the works of Shaw *et al.* [37], the research suggests that it is possible to examine other "personality traits" as potential threat indicators.

The Schultz preliminary framework even suggests a way to quantify all these metrics by means of a multiple regression equation that consists of the summation of the ITQA metric variables multiplied by their weightings. If $X_1, X_2, X_3 \cdots X_N$ represent the quantified ITQA metrics, $W_1, W_2, W_3 \cdots W_N$ their respective weights and C an arithmetic offset constant, then the expected estimated threat X_e is derived by:

$$X_e = (\Sigma W_i X_i) + C = W_1.X_1 + W_2X_2 + W_3X_3 + \cdots + W_NX_N + C$$

One notable absence of the Schultz insider threat prediction scheme is that there is no direct association between the estimated level of threat and the legitimate user's level of technical knowledge. Although the proposed metrics can provide evidence

that could be used to infer the level of user sophistication, there is no mentioning of a mechanism that takes that into consideration. Given the fact that, at the time of writing, the field of Insider Threat modelling is premature to reveal any usable results, it is difficult to prove the real impact of user sophistication on the threat level. On the other hand, Wood's model, a number of case studies and the survey results (Section 1.1) provide strong indications that there is a direct relationship between these two concepts. In that sense, the lack of a legitimate user sophistication gauging component could present a serious omission of the Schultz framework.

In addition, the exploitation of future mechanisms that will associate personality traits to potential misuse threat levels raises certain ethical and feasibility concerns. It is outside the scope of the thesis to examine ethical issues and the various laws that are associated with them. Nevertheless, the process of designing a model that is going to be employed in the real world should take into consideration its troublesome aspects. A metric that penalises real people in terms of their character traits will be considered unethical by many and depending on regional legislation may be also not feasible to implement.

In summary, the Schultz framework is more refined than Wood's earlier Insider Threat model in that it provides more concrete examples of ITQA metrics as well as a basic quantification mechanism for them. However the framework is still in its infancy. The author acknowledges that the chosen metrics need further refinement in order to prove their usefulness in a threat estimation process.

Brancik's [6] seminal work on the insider threat modelling should be referenced as a good source of information. Brancik's efforts center around information alteration, which is an important element of insider fraud, despite the fact that insider misuse surveys indicate that the frequency of these incidents are lower than other most common misuse incidents (web and email abuse). His Tailored Risk Integrated Process (TRIP) is the most important contribution. However, a risk management process deviates from traditional modelling approaches. This is because it focuses on factor evaluation. Detection of threat metrics is not addressed extensively.

Finally, all of the aforementioned research efforts do not address the issue of managing the representation of the data that feed the model component functions. One could argue that a preliminary model design needs to focus more on the scope, quality and quantity of its insider threat modelling functions. On the other hand, a well-thought definition of the procedures that represent and store the data that feed the threat modelling functions may have a notable impact on the computational efficiency and acceptance of the model. The reasons that support the need for this requirement are going to become apparent in the following paragraphs.

For all these reasons, we need a more formalised and broader model description. We have earlier published an Insider Threat Prediction Model that attempts to overcome the shortcomings of previous research work [18].

Considering a legitimate user population that has access to various components of an IT infrastructure, the core of the Insider Threat Prediction Model is a three-level hierarchy of mathematical functions evaluated in a bottom-up approach. At the top level, the Evaluated Potential Threat (*EPT*) function provides an integer value that quantifies and classifies the potential threat for each legitimate user into three

$$EPT = FITPQA = F_{attributes} + F_{behavior}$$

$$EPT = C_{role} + F_{accessrights} + F_{behavior}$$

$$EPT = C_{role} + C_{sysadm} + C_{criticalfiles} + C_{utilities} + C_{physicalaccess} + F_{sophistication} + F_{fileops} + F_{netops} + F_{execops}$$

Fig. 7 The Magklaras and Furnell model equation.

different categories. If x denotes the computed *EPT* for a legitimate user, *EPT_MAX* a threshold *EPT* value for considering the user a threat and *EPT_MIN* a threshold *EPT* value for considering the user's on line presence as suspicious, then:

- Important internal threat ($x = EPT\_MAX$): It indicates a high potential of a particular user misusing the system.
- Suspicious ($EPT\_MIN \leq x < EPT\_MAX$): This flags a condition where a particular user behaves in a manner that does not constitute a substantial threat but it is still a concern.
- Harmless ($0 \leq x < EPT\_MIN$): To indicate that the potential of misuse is nearly non existent for a particular user.

Each of the threat component functions models particular aspects of insider attributes and behavior. At the moment, in order to devise a well structured organization of threat components, the suggestion is to provide two threat component functions. The first one considers legitimate user attributes such as access rights and professional role, whereas the second evaluates potential threat simply by examining aspects of user behavior at the system level, as shown in Figure 7.

Table 1 lists the maximum weights of the nine top-level *EPT* formula components shown in Figure 7. Some of these components are constants (C_{role}, C_{sysadm}, etc) that belong to the $F_{attributes}$ function, whereas others constitute sub-functions of the $F_{behavior}$ function that address the assessment of the legitimate user on-line behavior.

It should be emphasized that the proposed maximum weights on Table 1 are not meant to be fixed. A system administrator/security specialist can re-define the maximum weights, in order to reward a particular metric that he trusts more than the others. For this reason, the nine weights of Table 1 constitute the *Weight Matrix*, a very important concept of the ITPM system. The Weight Matrix allows a specialist to further tune the sensitivity of the model, depending on the way he constructs misuse signatures, his confidence on the various metrics and the nature of the incident he is trying to predict. This feature enhances the adaptability of the proposed model scheme and enables to represent decision theoretic information.

The reader can refer to [22] for more details of the model and the reasoning behind the design of the proposed threat estimation functions. Two important things from this model should be emphasize here:

- the inclusion of various ITQAs at various levels (file, network, process execution) to represent a variety of system detectable user events.

EPT Component	Maximum Weight	Meaning
C_{role}	6	What is the documented role of the user inside the organization?
C_{sysadm}	6	Has the user access to Operating System administration utilities?
$C_{criticalfiles}$	6	Is it meant for the user to access commercially sensitive files?
$C_{utilities}$	6	Can the user execute application critical utilities?
$C_{physicalaccess}$	6	Has the user physical access to critical parts of the IT infrastructure?
$F_{sophistication}$	10	How capable is the user in terms of his computer system knowledge?
$F_{fileops}$	20	What are the signs of forthcoming insider misuse at file-level?
F_{netops}	20	What are the signs of forthcoming insider misuse at data network level?
$F_{execops}$	20	What are the signs of forthcoming insider misuse at program execution level?

Table 1 A sample Weight Matrix in the Magklaras and Furnell model.

- The introduced Weight Matrix concept as a mechanism of expressing different levels of confidence for the various ITQAs for a particular threat description.

Both of these things play a great role in the design of the ITPSL. The next section will explore the ITPSL relationship to the threat model process, as well as the overall scope of its inception.

4 The Scope of the Insider Threat Prediction Specification Language

Information security surveys and notable insider misuse cases reported by mass media were discussed in Section 1.1 of this report. The earlier paragraphs of introduced a more systematic presentation of the insider misuse domain by presenting a suitable insider taxonomy and a resulting insider threat model. However, how a threat model fits with a threat description language is not very clear. Figure 8 illustrates the relationship of the ITPSL and the proposed ITPM model.

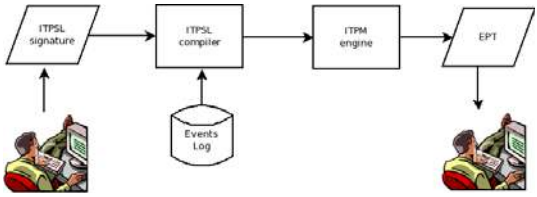

Fig. 8 The relationship between ITPSL and ITPM.

The flow of information starts with a security analyst writing a description of the particular insider misuse scenario, using the ITPSL semantics. The signature is validated by a compiler that translates the signature directives to query commands and makes use of an event logging infrastructure, in order to examine whether the ITQAs the signature mentions exist in the system. Apart from the semantics that qualify/quantify the ITQAs, the signature embodies a Weight Matrix statement which indicates the confidence of each specified ITQA. The results are passed to the ITPM engine which then derived an EPT value, indicating that a likelihood of a particular threat.

Figure 8 also includes the security analyst/system specialist both at the beginning of the information flow (signature construction) and at the final stage, where the final assessment is done. This emphasizes that the analyst is in charge of the process, both in terms of defining what constitutes a threat and also in terms of judging whether the likelihood expressed by the model is accurate.

This places the foundation of the context of ITPSL as a component of an entire Insider Threat management architecture [22]. The model estimates a threat which is described by a language and the likelihood is judged by the IT specialist. The emphasis is on the description and thus, the language addresses the lack of case repositories that express details of insider misuse incidents is apparent. An early report outlining aspects of the insider threat to the US government information systems published by the NSITSSAM Committee [27] considers the absence of case repositories as one of the limiting factors in the field of insider IT misuse mitigation research. In addition, the Carnegie Mellon University CyLab's 'Common Sense Guide to Prevention and Detection of Insider Threats' publication [8] states clearly the need to keep detailed records of employee actions in relation to file access, application usage and network connection matters.

ITPSL could also be a tool for digital forensic investigators. Digital forensics is an important research discipline of the information security field that is concerned with providing evidence to legal proceedings by means of gathering data to determine exact details of various types (internal and external origin) of system attacks. Brancik [6] mentions the importance of suitable tools to produce Key Fraud Signatures (KFS) to aid insider threat mitigation and thus signifies the overlap between insider misuse and the field of digital forensics.

The most widely used form of data forensic investigation is quiescent or static analysis. For such type of analysis, an investigator would utilize a number of toolkits to make a forensically valid copy of the affected system's non-volatile data storage

media and perform a "post-mortem" examination of the copied media. The goal is to examine static data (documents, images, email and system files) for digital evidence. AccessData's Forensic Toolkit [1] and Guidance Software's Encase [19] are two well known toolkits that perform, amongst other things, static digital forensic analysis.

However, static digital forensic analysis reveals an incomplete picture of the system in question. It cannot portray accurately the non-quiescent (dynamic) state of the system under investigation. Information such as active network endpoints, running processes, encryption keys for decrypted on-disk content, user interaction data (number of open applications per user, exact commands), as well as the content of memory resident processes may not be recorded accurately on non-volatile media. Hay *et al.* [20] discusses the shortcomings of static digital forensics analysis in detail. In order to overcome the barriers of static analysis, Adelstein [2] discusses the virtues of non-quiescent or live analysis, which essentially gathers data while the system under-investigation is operational. Microsoft's Computer Online Forensic Evidence Extractor (COFEE) [25] and FATkit [29] are two examples of tools that are able to extract live forensic data from volatile storage locations of a computer system.

Live data forensics analysis fills the gap of static examination methods, but it has its own disadvantages. Carrier [9] and Hay *et al.* [20] discuss the risks associated with acquiring live digital forensic data. In particular, live analysis methods suffer from three basic problems:

1. Investigator privileges: The investigator needs administration or escalated privileges to run the live analysis utilities. This could present a number of problems in environments where access policies prohibit escalated privileges from external entities to computer systems.
2. System and data integrity:The data gathered during the live analysis phase might be compromised due to system (due to rootkit infection, misconfiguration or intentional alteration of data by one or more system users). Whilst the memory data retrieval issue has been addressed by some complex hardware configurations whose purpose is to reliably acquire volatile data from system memory, the rest of the data acquisition issues are serious and they stem from the fact that data are logged on the system under investigation and not on a safer area before they are analyzed.
3. The "observer effect": When static analysis methods are used, the investigator can examine the data without affecting the source media state. Unfortunately, that is not true for live analysis where the investigator's actions can affect the data. One would have to separate carefully the implications of the investigator's actions from the original data.

The previously mentioned needs shape the scope of the Insider Threat Prediction Specification Language (ITPSL): A specialized language that is able to encode system level data that concern legitimate user actions, in order to aid the process of misuse threat prediction and assist computer forensic officers in the process of examining insider misuse incidents. As such, ITPSL's target audience is the security

analyst/expert, as well as the seasoned IT administrator in charge of system operation and security issues. Both of these types of domain experts should be able to express insider misuse scenarios by using the language semantics to construct signatures of threat scenarios. More specifically, the ITPSL language should be able to meet the following high level functional requirements.:

- FR1: Separate the analysis data from the target system(s) to minimize issues with maliciously or accidentally altering the data.
- FR2: The architecture of the language should facilitate the creation of suitable insider threat signature repositories, so that security specialists/system administration could easily browse for signatures of various threat scenarios. This feature aims to address the lack of suitable case repositories discussed in the earlier paragraphs of this section.
- FR3: Its semantics and logging mechanisms should facilitate the description of both static and live forensic insider misuse system data at the network, process and filesystem layer, in response to the issues discussed .
- FR4: The semantic description of user actions should encompass temporal indicators so that sequences of events could clearly be expressed and logged.
- FR5: The language should be able to represent decision theoretic information to address the criticisms of earlier intrusion specification examples such as CISL [13]. This implies the ability to consistently express various potentials scenarios of insider actions, giving the signature polymorphic properties.
- FR6: The semantics of the language should offer a consistent hierarchical way of describing a variety of scenarios and should be easily readable by humans and software modules.
- FR7: Finally, ITPSL should have an operating system agnostic scope. The signature author should use the same semantics to express the various misuse threat scenarios regardless of whether the underlying operating system is Microsoft Windows, Linux/Unix, MACOSX or other applicable platform. The language semantics should bridge any gaps created by operating system esoteric peculiarities that could affect the process of expressing threat indicators.

4.1 The Domain Specific Language Programming Paradigm

The ITPSL scope defines clearly a specific task of expressing insider threat metrics. This paves the way for the selection of a mechanism that allows the language designer to focus on the problem in question. A Domain Specific Language (DSL) is a semantic mechanism tailored specifically for describing the details of a particular task. The main goal is the usage of appropriate semantics to reduce the effort required to reference and manipulate elements of that particular domain.

Spinellis [39] defines a Domain Specific Language as “programming language tailored specifically to an application domain: rather than being for a general purpose, it captures precisely the domain’s semantics”. DSL schemata have been employed successfully in a number of different areas. Consel [11] discusses the range

of applications that have employed a DSL which includes device driver construction, active networking and operating system process scheduling. Moreover, Raymond [33] outlines some widely known 'mini' languages employed in the Unix community (regular expressions, awk, m4) and beyond (Postscript, SNG, Glade) as examples of domain specific languages. This list is by no means exhaustive, as many more DSLs exist today. A DSL is really a framework that offers the ability of building specific and concise notations to express a problem domain, as well as safe (as predictable) code due to semantic restrictions. Both of these properties are very desirable in the process of developing insider threat specifications.

DSLs are also categorized as external and internal in terms of the way they are implemented [21]. External DSLs are discrete systems, independent from any host language and they contain their own interpreter or compiler to parse the language statement and perform post interpretation/compilation actions. In contrast, internal DSLs are semantics embedded inside a general purpose programming language and thus are dependent from the interpreter/compiler of the host language. Examples of external DSLs are the 'mini' Unix languages, whereas internal DSL languages tend to be embedded in programming anguages such as Lisp, Smalltalk, and Ruby.

The process of deciding which DSL approach to follow for implementing ITPSL is important. External DSL approaches offer a greater freedom to experiment with the process of constructing insider threat semantics but they provide a higher overhead when it comes to development issues combined with a higher learning curve for the language users. On the other hand, internal DSLs offer less development overhead as parsing, interpretation and compilation issues are handled by the host language environment. If one takes into account that the host general programming language will have already mature semantics and an established user base, it is easy to conclude that an internal DSL would have less steep learning curve than an external DSL approach.

However, the internal DSL dependency on the host language environment might create problems for the language designer. The most important issue might arise from a mismatch between the symbolic integration of the embedded DSL and the general vocabulary of the general purpose host language. General purpose language vocabularies are rich enough to express a variety of scenarios in an abstract way. For example, on a network access scenario, a general purpose programming language vocabulary can express details of the origin and destination of a network connection but not express network access patterns. In that case, if one tries to engineer the additional functionality into the general language, the process of constructing meaningful semantics might be impaired due to the general language syntax or due to the host language underlying data structures that might not be able to represent fully the required domain information.

A secondary practical problem of adopting an internal DSL approach might include parameter evaluation and performance issues. An insider threat prediction operational environment requires the evaluation of various parameters at runtime. If a statically compiled host general language is used (such as C/C++), runtime evaluation of parameters might pose a challenge. There are of course scripting languages [28] where runtime evaluation is not an option, but they might be slow.

Ways to combine compile and runtime languages do exist (*i.e.*, a Perl Script calling a C/C++ library via API wrappers), however the complexity of combining domain specific semantics with more than one language should not be underestimated.

For all these reasons, ITPSL follows the external DSL approach allowing for freedom to create the semantics from scratch with commonly changed parameters to be altered without recompilation issues and no dependence on host language idiosyncrasies. The issue of the learning curve for a domain expert to learn yet another language is of course considerable. However, the narrow scope of a DSL language combined with carefully crafted semantics should create a low complexity interface of relatively few (when compared to a general purpose language) statements and thus make the language easy to learn. This approach has been followed by a number of security related research DSLs such as CISL [13] and Panoptis [40], as discussed in previous sections. For now, it should be noted that both of them can be categorized as external DSLs using configuration files to encode statements that have no resemblance to general purpose programming languages.

There are also a number of external DSLs that utilize XML to convey information. Using XML as a markup to construct DSLs is a common approach and thus XML-enabled DSLs are the subject of the next section.

5 Conclusion

The ability to specify insider threat scenarios can be a useful threat mitigation technique. Starting with suitably crafted insider misuse taxonomies, we develop a standardized vocabulary to describe system-level aspects of insider threat scenarios. This qualifies suitable Insider Threat indicators. Insider Threat models help to quantify the threats and provide a measure of the likelihood of the occurrence of a particular misuse scenario. Finally, a Domain Specific Language Insider Threat Prediction Language (ITPSL) designed to describe these scenarios will be the focal point of a threat mitigation technique. Such a language could also complement forensic tools, acting as a repository of events capable of replaying certain threat scenarios.

The design and construction of ITPSL is a work in progress.

References

1. Accessdata inc. web portal, available at `http://www.accessdata.com`, last accessed in march 2010.
2. Adelstein, F.: Live forensics: diagnosing your system without killing it first. Communications of the ACM **49**(2), 63–66 (2006). DOI http://doi.acm.org/10.1145/1113034.1113070
3. Amoroso, E.G.: Intrusion detection : an introduction to Internet surveillance, correlation, traps, trace back, and response, 1st ed edn. Intrusion.Net Books, Sparta, NJ (1999)
4. Bach, M.J.: The design of the UNIX operating system. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1986)

5. Bishop, M., Gollmann, D., Hunker, J., Probst, C.: Countering insider threats. In: Dagstuhl Seminar 08302, Dagstuhl Seminar Proceedings, p. 18 pp. Leibnitz Center for Informatics (2008)
6. Brancik, K.: Insider Computer Fraud; An Indepth Framework for Detecting and Defending Against Insider IT Attacks. Auerbach Publications, Boston, MA, USA (2007)
7. Caelli, W., Longley, D., Shain, M.: Information Security Handbook. Stockton Press (1991)
8. Cappelli, D., Moore, A., Shimeall, T., R., T.: Common sense guide to prevention and detection of insider threats. Tech. rep., Common Sense Guide to Prevention and Detection of Insider Threats (2006). Available at http://www.cert.org/archive/pdf/CommonSenseInsiderThreatsV2.1-1-070118.pdf, last accessed March 2010.
9. Carrier, B.D.: Risks of live digital forensic analysis. Communications of the ACM **49**(2), 56–61 (2006)
10. The common intrusion detection framework (CIDF). Available at http://gost.isi.edu/cidf, last accessed in March 2010.
11. Consel, C.: From a program family to a domain-specific language. In: Domain-Specific Program Generation, International Seminar, Dagstuhl Castle, Germany, March 23-28, 2003, pp. 19–29 (2003)
12. Doyle, J.: Some representational limitations of the common intrusion specification language. Tech. rep., Laboratory for Computer Science, Massachusets Institute for Technology, Cambridge MA (1999)
13. Feiertag, R., Kahn, C., Porras, P., Schnackenberg, D., Staniford-Chen, S., Tung, B.: A Common Intrusion Specification Language (CISL) (1999). Available from http://gost.isi.edu/cidf/drafts/language.txt, last accessed in March 2010.
14. Frykholm, N.: Countermeasures against buffer overflow attacks. Tech. rep., RSA Laboratories (2000)
15. Furnell, S., Magklaras, G., Papadaki, M., Dowland, P.: A generic taxonomy for intrusion specification and response. In: Proceedings of Euromedia 2001, pp. 125–131 (2001)
16. Furnell, S., Papadaki, M., Magklaras, G., Alayed, A.: Security vulnerabilities and system intrusions - the need for automatic response frameworks. In: Proceedings of the IFIP TC11 WG11.1/WG11.2 Eigth Annual Working Conference on Advances in Information Security Management & Small Systems Security, pp. 87–98. Kluwer, B.V., Deventer, The Netherlands (2001)
17. G., M., S., F.: The insider misuse threat survey: investigating it misuse from legitimate users. In: Proceedings of the 5th Australian Information Warfare & Security Conference, pp. 42–51 (2004)
18. G., M., S., F.: A preliminary model of end user sophistication for insider threat prediction in it systems. Computers & Security **24**(5), 371–380 (2005)
19. Guidance software inc. web portal, available at http://www.guidancesoftware.com, last accessed in march 2010.
20. Hay, B., Bishop, M., Nance, K.L.: Live analysis: Progress and challenges. IEEE Security & Privacy **7**(2), 30–37 (2009)
21. M., F.: Language workbenches: The killer-app for domain specific languages? Available from http://martinfowler.com/articles/languageWorkbench.html, last accessed in March 2010.
22. Magklaras, G.: An architecture for insider misuse threat prediction in it systems. Master's thesis, School of Computing, Communications and Electronics, University of Plymouth, UK (2005)
23. Magklaras, G., Furnell, S., Brooke, P.J.: Towards an insider threat prediction specification language. Information Management & Computer Security **14**(4), 361–381 (2006)
24. McAuliffe, W.: Firms shop around for net law jurisdictions (2001). Available at http://news.zdnet.co.uk/itmanagement/0,1000000308,2085983,00.htm, last visited March 2010.
25. Microsoft Corp.: The computer online forensic evidence extractor (cofee). Available online at http://www.microsoft.com/industry/government/solutions/cofee, last accessed March 2010.

26. Moore, D., Voelker, G.M., Savage, S.: Inferring internet denial-of-service activity. In: SSYM'01: Proceedings of the 10th conference on USENIX Security Symposium. USENIX Association, Berkeley, CA, USA (2001)
27. The insider threat to us government information systems. Tech. rep., U.S. National Security Telecommunications And Information Systems Security Committee (1999). Available `http://www.cnss.gov/Assets/pdf/nstissam\_infosec\_1-99.pdf`, last accessed March 2010.
28. Ousterhout, J.K.: Scripting: Higher-level programming for the 21st century. Computer **31**, 23–30 (1998). DOI http://doi.ieeecomputersociety.org/10.1109/2.660187
29. Petroni, N.L., Aaron, J., Timothy, W., William, F., Arbaugh, A.: Fatkit: A framework for the extraction and analysis of digital forensic data from volatile system memory. Digital Investigation **3**(4), 197–210 (2006)
30. Pfleeger, C.P., Pfleeger, S.L.: Security in Computing (4th Edition). Prentice Hall PTR, Upper Saddle River, NJ, USA (2006)
31. Postel, J., Reynolds, J.: TELNET Protocol Specification, Request For Comments (RFC) 854. IETF Network Working Group (1983)
32. Power, R.: 2001 csi/fbi computer crime and security survey. Computer Security Journal **17**(2), 29–51 (2001)
33. Raymond, E.: The Art of UNIX Programming. Addison-Wesley Professional (2003)
34. Richter, J.: Advanced Windows. Microsoft Press, Redmond, Washington, USA (1997)
35. Schultz, E.E.: A framework for understanding and predicting insider attacks. Computers & Security **21**(6), 526–531 (2002)
36. Sharda, N.K.: Multimedia information networking. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1998)
37. Shaw, E., Ruby, K., Post, J.: The insider threat to information systems. Security Awareness Bulletin **98**(2) (1998)
38. Sommerville, I.: Software engineering (5th ed.). Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA (1995)
39. Spinellis, D.: Notable design patterns for domain-specific languages. Journal of Systems and Software **56**(1), 91–99 (2001)
40. Spinellis, D., Gritzalis, D.: Panoptis: intrusion detection using a domain-specific language. Journal of Computer Security **10**(1-2), 159–176 (2002)
41. T., A., I., K., E., S.: Use of a taxonomy of security faults. Tech. Rep. TR-96-051, COAST Laboratory, Department of Computer Sciences, Purdue University (1996)
42. Tugular, T.: A preliminary structural approach to insider computer misuse incidents. In: Proceedings of the EICAR Conference 2000, pp. 105–125. European Institute for Computer Antivirus Research (EICAR) (2000)
43. Wood, B.: An insider threat model for adversary simulation. In: Proceedings of the Workshop on Research on Mitigating the Insider Threat to Information Systems (2000)
44. Ylonen, T.: The SSH (Secure Shell) Remote Login Protocol, Internet Draft. IETF Network Working Group (1995). Available `http://www.free.lp.se/fish/rfc.txt`, last accessed March, 2010.
45. Ziegler, R.: Linux firewalls. New Riders Publishing, Indianapolis, IN, USA (2002)