DEVELOPMENT OF A METHODOLOGY FOR CUSTOMIZING INSIDER THREAT AUDITING ON A MICROSOFT WINDOWS XP® OPERATING SYSTEM

THESIS

Terry E. Levoy, Gunnery Sergeant, USMC

AFIT/GIA/ENG/06-07

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the U.S. Government.

II. AFIT/GIA/ENG/06-07

DEVELOPMENT OF A METHODOLOGY FOR CUSTOMIZING INSIDER THREAT AUDITING ON A MICROSOFT WINDOWS XP® OPERATING SYSTEM

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science

Terry E. Levoy, BBA

Gunnery Sergeant, USMC

May 2006

AFIT/GIA/ENG/06-07

DEVELOPMENT OF A METHODOLOGY FOR CUSTOMIZING INSIDER THREAT AUDITING ON A MICROSOFT WINDOWS XP® OPERATING SYSTEM

Terry E. Levoy, BBA

Gunnery Sergeant, USMC

Approved:

//SIGNED//

_____ _____

Dr. Robert F. Mills, (Chairman) Date

//SIGNED//

_____ _____

Dr. Michael R. Grimaila, (Member) Date

//SIGNED//

_____ _____

Dr. Gilbert L. Peterson, (Member) Date

//SIGNED//

_____ _____

Mr. Timothy H. Lacey, (Member) Date

Abstract

Most organizations are aware that threats from trusted insiders pose a great risk to their organization and are difficult to protect against. Auditing is recognized as an effective technique for detecting malicious activity. However, current auditing methods are typically applied with a one-size-fits-all approach and may not be an appropriate mitigation strategy for the insider threat.

This research develops a 4-step methodology for designing a customized auditing template for a Microsoft Windows XP® operating system. Two tailoring methods are presented which evaluate both by category and and by configuration. Also developed are various metrics and weighting factors as a mechanism to evaluate auditing effectiveness for the purpose of optimizing the template according to organizational security requirements. Various industry standard auditing templates are evaluated against a custom designed template. Results indicate that a customized auditing template tailored for an insider threat scenario is more effective at detecting insider malicious activities.

Acknowledgments

First and foremost, I would like to give my sincere appreciation to my wife, who has been by my side in everything I've done. I am truly blessed to have her in my life. Secondly, I thank my three sons for giving me motivation. Their jovial attitudes, quick wit, and occasional hugs have made it easier to cope with the stressful times. I am thankful for my entire family putting up with my periods of absence and extreme grouchiness in this endeavor.

I would like to thank my advisor, Dr. Robert Mills, for helping me with getting on the right track with my thesis topic, and for giving me enough guidance and latitude to accomplish this challenging feat. I would also like to thank Dr. Michael Grimaila, who has selflessly contributed help and guidance including assistance with the programming and metrics needed for this thesis. I would also like to thank Dr. Gilbert Peterson for his help with the set theory, and Mr. Tim Lacey for always being there with a friendly smile and a precise answer when I needed technical help.

I am deeply indebted to my friend, fellow Marine, and AFIT graduate, MSgt Juan Lopez. He not only helped me with the acceptance process into AFIT, but also continuously helped me along the way. He has been there for me right up to the end.

Terry E. Levoy

Table of Contents

Page

List of Figures

List of Tables

DEVELOPMENT OF A METHODOLOGY FOR CUSTOMIZING INSIDER THREAT AUDITING ON A MICROSOFT WINDOWS XP® PROFESSIONAL OPERATING SYSTEM

I. Introduction

This research introduces a methodology for developing a customized auditing template for a computer workstation. The methodology is designed to tailor the auditing settings for a workstation using the Microsoft Windows XP® Operating System according to the unique organizational security requirements in regards to an insider threat.

1.1. Research Motivation

Information has always been a critical resource and key ingredient for decision-making abilities. The rapid growth of microcomputers and distributed networks has significantly increased the amount of information available to decision makers in electronic form. One drawback of the availability of widespread information in electronic form has been the inherent complications of protecting that information. The protection of information is an extremely critical area of study.

Threats to an organization's information resources can be classified in three categories: (1) outside threats, (2) inside threats, and (3) natural threats. Most organizations are aware of threats to their information resources and do what they can to protect them. Natural threats such as natural disasters and power losses are generally relatively simple to understand and require planning and procedures to mitigate, such as backup strategies. Outsider threats are more complicated and are commonly countered with technological solutions such as firewalls, Intrusion Detection Systems, and antivirus

software. The threat from trusted insiders can be significant to an organization, and significantly more difficult to mitigate. One approach to deal with the insider threat is to implement measures to detect the malicious actions that they perform. One technique to accomplish this is by auditing their actions. If this auditing is properly performed, then insider activities that stray from the norm can be recognized and investigated. One resource available to accomplish this is by invoking the auditing capabilities available in Microsoft Windows XP®. The Windows XP® Operating System provides a wide variety of auditing capabilities which can be used to detect potential malicious insider activities. Although these auditing capabilities exist, they are not enabled by default. An organization needs to configure client workstations to audit events based on its identified security requirements. Determining the events to audit is a difficult process. Currently, auditing configurations that are available are inherently tied to one-size-fits-all security templates. (e.g. *NIST EC*, *NIST SSLF*, *NSA SNAC*, etc.) There exists a need for a method which assists an organization in tailoring its auditing based on its unique security requirements.

1.2. Problem Statement

Organizations are aware of threats to their information resources, and spend countless hours and money attempting to protect them. Most of their efforts are concentrated on countermeasures aimed at protecting against threats from outside the organization, but organizations often overlook protecting information against the insider threat. The threat from malicious insiders is substantial, and worth serious consideration. One strategy to protect against malicious insiders is detecting their activity through

auditing. This is usually done within organizations, and usually blindly through the use of preexisting auditing configurations. Currently, there exists no defined methodology for creating a customized auditing template based on organizational security requirements.

1.3. Assumptions

In order to use the methodology developed in this study, an organization needs to have identified security requirements so that it can properly address them. This is accomplished by performing a risk analysis. Before applying the results of this thesis, it is assumed that a risk analysis has been accomplished already.

1.4. Scope

There are currently many existing threats to an organization's information infrastructure. The focus of this study is on the insider threat because it is difficult to mitigate. Auditing can be extremely effective at detecting insider threats if configured properly. This study is limited to the use of local workstation auditing with local users only.

1.5. Overview

This research has developed a 4-step methodology which when used to creates a customized auditing template that is more beneficial than using any preexisting auditing templates available to support security configuration templates. It includes the steps required for setting up and conducting this process. The methodology includes two separate methods for creating an auditing template and provides a method for evaluating each.

1.6. Thesis Overview

This chapter defines the research goal and provides a brief summary of the motivation for developing an organizational customized auditing template. Chapter 2 presents a literature review and provides background information that supports this research. Chapter 3 describes the methodology, experimental setup, and calculations used to conduct this research. Chapter 4 presents the experiment results and an analysis of those results. Chapter 5 provides conclusions from this research and recommendations for additional research in the area of creating auditing templates.

II. Literature Review

2.1. Chapter Overview

This chapter presents the concepts and current trends of insider threats. It also presents the current uses of audit logs to monitor workstation activity inside an enterprise network to enable an organization to recognize insider threat indicators. The first section presents the objectives of information security. The second section offers a definition of insider threat including its characteristics and types. It then discusses current trends in the area of detecting and mitigating the insider threat. Then various insider attack methods and the ways of identifying those methods are discussed. The third section summarizes the history of auditing to monitor and identify insider behaviors and its potential role in identifying the characteristics of insider threats. The final section discusses the current uses of auditing within the workstation environment, its capabilities, and its various implementations.

2.2. Objectives of Information Security

Protecting an enterprise network against both outside and inside attacks involves balancing information system usability with three main security objectives: confidentiality, integrity, and availability (*CIA*). The National Institute of Standards and Technology (*NIST*) describes these three objectives in the Federal Information Processing Standards Publication (*FIPS PUB*) 199, which were originally established as part of the Federal Information Security Management Act of 2002 in addition to providing guidance on the categorization of information systems. [1, 2]. These three main objectives are ideal as the main focus areas that an organization should consider when protecting its

information resource assets, however they need to be balanced with usability. It is important that information owners within organizations understand these objectives so they can better achieve a balance between the security of their organizations information and its useability. Now that we have an understanding of the need for the security objectives, we define them.

2.2.1. Confidentiality

Confidentiality is "preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information… [1]" It refers to the privacy of information, or more specifically which individuals, entities, or processes are authorized access to that information at what times [3]. Access includes viewing, processing, displaying, processing, and transmitting the information. For example, at a medical facility, only patients and medical staff, who require access to perform their jobs, can access patient medical records. Ensuring confidentiality is critical to an organization because the loss of confidentiality, which can lead to the unauthorized disclosure of critical or sensitive information, can lead to severe economic losses or loss of life within an organization [1, 4].

2.2.2. Integrity

Integrity is the assurance that "data has not been changed, destroyed, or lost in an unauthorized or accidental manner" [3]. It means that trusted information is expected to be modified only by duly authorized individuals. For example, in an educational institution only authorized staff, such as the registrar, should be able to modify a student's grades, and a professor should be able to view but not modify them. Ensuring

integrity within an organization is also critical because the loss of integrity, and the ensuing unauthorized modification or destruction of information, can result in the serious loss of organizational effectiveness, or serious economic loss [1, 4].

2.2.3. Availability

Availability is "ensuring timely and reliable access to and use of information and information services…" [1] The loss of availability results in the disruption of access to or use of information or an information system [1, 2]. Information must be available to authorized persons, entities, or devices whenever they are required according to the information system design [3]. Normally, a premium is put on preserving the availability of the most critical (or valuable) information. For example, even the temporary unavailability of a financial company's databases might result in losses of millions of dollars. Therefore, the company must put great importance on preserving the availability of its databases. An organization should consider allocating appropriate resources to protect the availability of resources based on information value to safeguard against loss of that information.

2.3. Insider Threat

Insider threat is an often misunderstood and overlooked phenomenon. It is substantially damaging to an organization and requires attention and understanding if it is to be protected against and mitigated.

2.3.1. Definition of Insider Threat

The following section defines insider threat and identifies its characteristics. In a 2004 National Threat Assessment Center Report done by the U.S. Secret Service and

CERT Coordination Center Carnegie Mellon University, a comprehensive definition of malicious insider threat was presented:

> Current or former employees or contractors who intentionally exceeded or misused an authorized level of access to networks, systems or data in a manner that targeted a specific individual or affected the security of the organization's data, systems and/or daily business operations [5].

The term *insider threat* can be determined to refer to the potential damage to an organization caused by the actions of one or more of its trusted agents. The activities causing damage can be intentionally malicious acts or simple user mistakes. Every organization should conduct a careful analysis of its insider threat situation and implement appropriate countermeasures to safeguard their information and information systems.

2.3.2. Insider Threat Trends

The recent impact of reported insider threat activities has been significant. In a 2005 Carnegie Mellon Insider Threat Study, it was found that eighty-one percent of organizations, who reported insider activity, experienced financial losses ranging from $500 to more that $10,000,000 [6]. These reported losses do not include implicit costs, such as lost future sales or loss of market capitalization, which are difficult or impossible to measure but devastating to an organization [7]. Figure 2.1 shows a trend of decreased reported network intrusions, which could be due to an increased awareness and more advanced technology to defend against malicious activity. It could also mean that organizations are reluctant to report intrusions, primarily because of the bad publicity and potential lost revenues associated with the bad publicity [6]. Figure 2.2 illustrates that in

2005 eighty-seven percent of organizations are conducting security audits, which is up five percent from 2004, which may indicate that they are becoming more aware of the benefits of these audits in helping to detect potential intrusions [7-9].

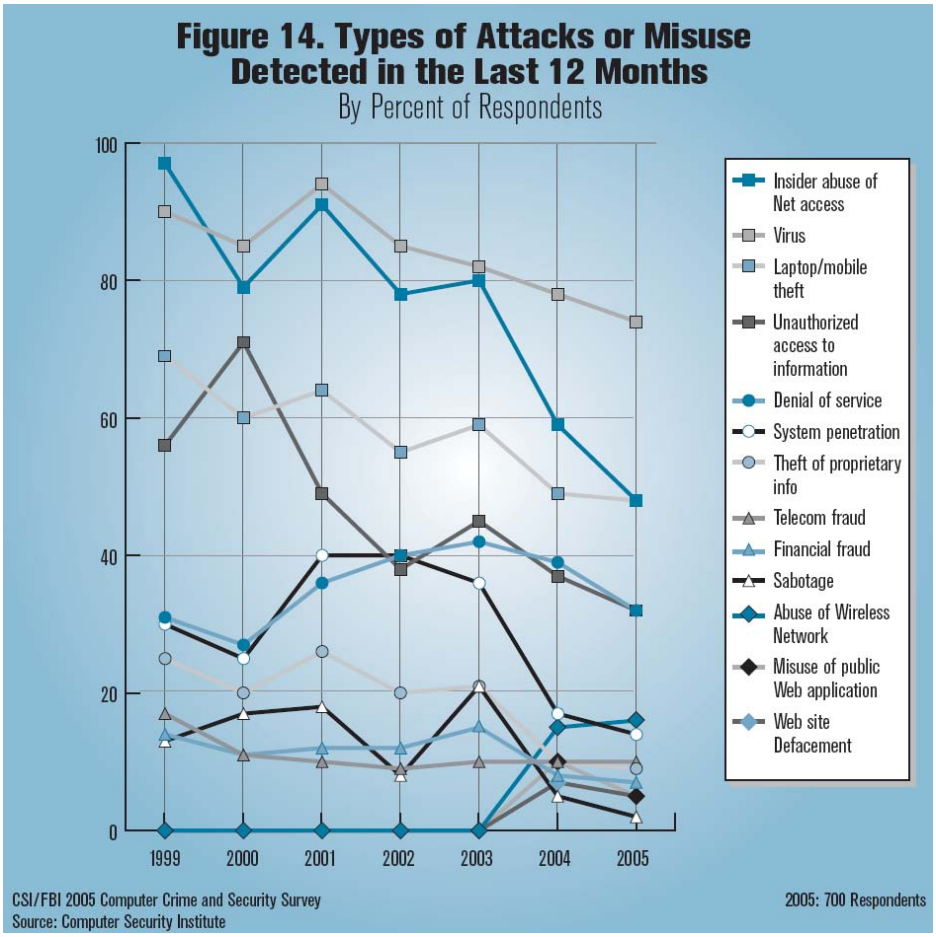

Figure 2.1: Attack trends [7]

2.3.3. Insider Threat Characteristics

Insiders have the potential to pose a significantly high threat to an organization because they are trusted agents. They have knowledge of, and access to, sensitive inside information, such as the organization's network design, security procedures, and organizational systems and databases, which potentially gives them opportunity and know-how to either damage or steal information [6]. They are also privy to information

about any security vulnerabilities the organization might have. Often they hold, or can easily gain, higher privileges such as administrative or super-user rights, which makes them considerably more capable of performing malicious acts. A recent study by Carnegie Mellon looked at case studies and discovered that 77% of the malicious insider activity was attributed to full-time employees, and 86% held technical positions. These technical positions included system administrators (38%), programmers (21%), engineers (14%), and IT specialists (14%) [6].

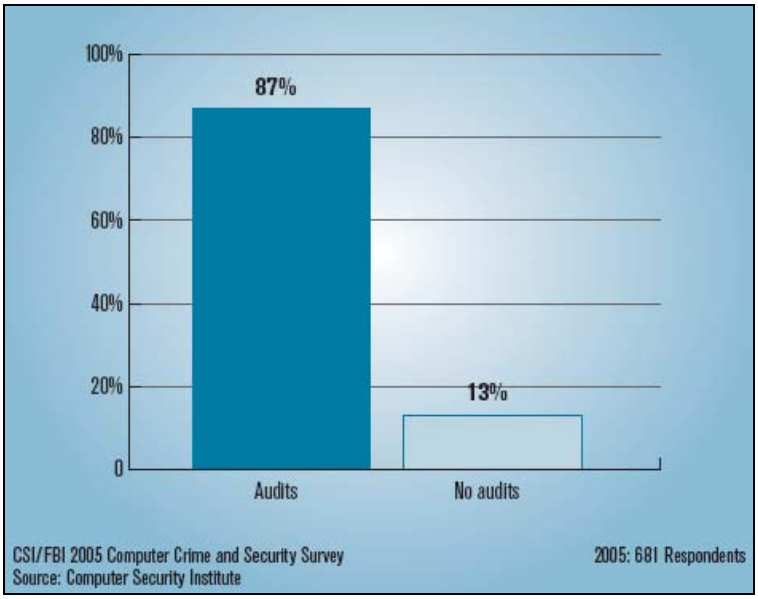

Figure 2.2: Organizations Conducting Security Audits [7]

2.3.4. Types of Insider Threats

The threat existing from insiders depends on the intent and activity of the insider and is classified by three types: normal, abnormal, and malevolent. Table 2.1 describes the three types of insiders, the activities that they exhibit, and their threat levels. Normal insider activities typically do not pose a threat because their activities consist of normal day-to-day use of organizational information resources without any abnormal behaviors

or situations. Abnormal insider activities are “out of the ordinary” events, and can include routine errors. These routine errors can render a system vulnerable, which can cause a breach in Confidentiality, Integrity, or Availability [10]. Examples of this are system administrators forgetting to delete accounts of users who have left the company, users leaving themselves logged into their workstations, or users using weak passwords for their accounts. These types of routine errors pose a substantial vulnerability to the information resources of an organization to intrusion by third parties that take advantage of the weaknesses exposed by the abnormal insider activities.

Malevolent insider activity is activity by individuals with malicious intent of stealing or damaging the information of the organization, which poses the greatest threat [10]. The motives of these insiders vary but are most often related to revenge. Negative work-related events often trigger these threats. In a recent study, United States Secret Service and CERT Program Software Engineering Institute of Carnegie Mellon University found that “in 92% of the cases, a specific event or series of events triggered the insider’s actions” [6]. These events include termination of employment (47%), dispute with a former employer (20%), and demotion or transfer (13%) [6]. An organization should know the characteristics of each of these types of insider threats in order to plan an appropriate mitigation strategy.

Table 2.1: Insider Activity Types

Insider Activity Type	Activity	Threat
Normal	Typical Behavior	Limited
Abnormal	Routine Errors	Substantial Threat
Malevolent	Malicious Intent	Greatest Threat

2.3.5. Attack Methods

The vulnerabilities most often exploited by an insider are the lack of, or the ineffectiveness of, controls and checks to prevent an insider from removing sensitive data from their work areas [11]. In a 2005 study conducted by the U.S. Secret Service and Carnegie Mellon's Computer Emergency Response Team Coordination Center found that a large majority of the reported incidents were not technically sophisticated. They found that 61% of the reported malicious insider actions were executed with no technical sophistication using simple attack methods such as legitimate user commands, information exchanges, or physical attacks [6]. In 60% of these cases, the insider compromised an account to carry out the attack, including the use of another user's username and password, or the use of an unauthorized account created by the insider.

These attacks were successful primarily due to systemic vulnerabilities in technology and/or policies, processes, or procedures, such as [6]:

- Coarse access control restrictions
- Sloppy accounting procedures
- Infrequent or non-existent monitoring procedures
- Insufficient physical access controls

These attacks were successful primarily due to systematic vulnerabilities in technology and/or policies, processes, or procedures, such as

- Scripts or malicious programs
- Autonomous agents
- Toolkits
- Flooding
- Probing
- Scanning
- Spoofing

Although malicious insiders usually use relatively uncomplicated methods in executing attacks, the very same lack of sophistication can actually make detection more difficult due to its resemblance to normal day-to-day activity in the organization. In order for organizations to detect malicious insiders, they need to identify any indicators that can alert them to possible insider actions.

2.3.6. Identifying Possible Insider Threats

The threats posed to an organization both from the inside as well as the outside have become more evident in recent years. Studies indicate that most organizations are becoming increasingly more aware of the importance and usefulness of technology used to combat malicious activity such as firewalls, intrusion detection systems, and antivirus solutions. In the 2005 CSI/FBI survey, 97% of the respondents reported using firewalls, 72% used intrusion detection systems, and 96% used antivirus software" [7]. These technologies mostly have perimeter defenses in mind, which are sufficient at detecting or preventing external attempts to access, by-pass, or damage organizational information or information systems. However, they do very little to detect insider threats. In a 2005 Carnegie Mellon University study, it was discovered that the majority of insider attacks were not detected by security personnel [6]. Furthermore, the malicious activity was only detected because of a noticeable change in the system performance, or because the

system simply became unavailable [6]. In the 2004 Carnegie Mellon E-crime Watch Survey, nearly half of the organizations reporting insider e-crimes reported that they were uncovered accidentally and not as a result of security policies [12]. These studies provide evidence to support the requirement to use more resources than perimeter defenses to identify attacks especially against insider threats. Organizations must develop effective policies and procedures to achieve better protection against the insider threat. One potentially useful and often overlooked method is the use of auditing.

2.4. Auditing

Auditing is a readily available resource that an organization can use to protect against the insider threat. First it must be understood, and its capabilities explored by investigating its background and various roles and uses.

2.4.1. Definition and Evolution of Auditing

Auditing, which originates from Latin meaning, “To hear”, has been in existence for ages [13]. The purposes of an audit are to provide an independent verification, reduce errors in record-keeping, reduce misappropriation of assets, and prevent or detect fraud. The roots of auditing are summarized by an accounting historian named Richard Brown who stated, “Whenever the advance of civilization brought about the necessity of one man being entrusted to the extent with the property of another, the advisability of some kind of check upon the fidelity of the former would become apparent” [13]. Historians believe that formal record-keeping systems were used to account for transactions as far back as 4000 BC [13]. Throughout history, people have used different forms of auditing. It was considered a prudent measure to ensure that trust was not broken, regardless of if it

was intentionally or not. As President Ronald Reagan stated in his farewell address, referring to our relationship with Russia after the end of the cold war, Ronald Reagan said, “It's still trust, but verify. It's still play, but cut the cards. It's still watch closely. And don't be afraid to see what you see” [14].

More recently, auditing has been identified as a critical element in the maintaining of information security. Auditing information systems is defined by the National Institute of Standards and Technology (*NIST*), the federal authority in information security standards and guidelines, as “the review and analysis of management, operational, and technical controls…. In which valuable information about activity on a computer system can be obtained” [15]. To keep information systems secure, the United States Federal Government has enacted the Computer Security Act of 1987. This act established minimum security practices for federal information systems to improve security and privacy. It also assigned *NIST*, with the assistance of the National Security Administration (*NSA*), the responsibility of developing standards and guidelines required to implement efficient security and privacy in federal information systems [16].

The *NIST* Special Publication 800-30, Risk Management Guide for Information Technology Systems, provides guidance on the use of auditing to assist in the assurance of an organization’s information and information systems. After security-related events, the evaluation of audit logs and the monitoring and tracking of abnormalities in systems are key elements for detecting and recovering from security breaches [17].

DoD instruction 8500.2 mandates that the collection and retention of all audit data be performed by all heads of DoD, as information owners, for the support of technical

analysis relating to misuse, penetration reconstruction, or other investigations [4]. This instruction also mandates that “an automated, continuous on-line monitoring and audit trail creation capability be deployed with the capability to immediately alert personnel of any unusual or inappropriate activity with potential Information Assurance (*IA*) implications…” [4]

2.4.2. Auditing Policy

In order to effectively use auditing to achieve security objectives, a well-developed auditing policy needs to be created. The baseline for implementing a good auditing policy is to start with a good security policy. An important part of the security policy is to identify the roles and responsibilities of individuals within the organization. If those roles and responsibilities are not identified, and their duties are not clearly defined, then identifying abnormal behaviors of individuals performing those roles through the use of auditing becomes increasingly difficult. The policy should also define the role of auditing, how auditing is performed, what activities need to be identified, and how particular anomalies will be recognized. Also, specific tasks need to be defined, such as who will review the logs, how often they will be reviewed, how long they will be stored, and appropriate reactions to actions found. In a recent study, CSO Magazine and Carnegie Mellon University Software Engineering Institute's CERT Coordination Center found that only half of organizations have a formal process or system in place for tracking e-crime attempts [12]. Without a formal process or system in place for tracking e-crime attempts, an organization may not even know if its network has been compromised.

2.4.3. The Role of Auditing in An Enterprise

Many occurrences happen within an organization's information infrastructure that can be recognized as potential insider threat activity. Most of these occurrences can be logged within the organization, and then observed if the audit logs are regularly reviewed by security administrators. As was presented by RAND in its 2004 workshop, there is a taxonomy of observables useable for recognizing characteristics of potentially malicious insiders (Figure 2.3). As seen in the center of the figure, most of the sub-categories which fall under the category Cyber Actions can be recognized through auditing because most are performed on information systems that have auditing capabilities. Auditing provides a means to accomplish these observations in several areas such as individual accountability, reconstruction of events, intrusion detection, and problem analysis [15].

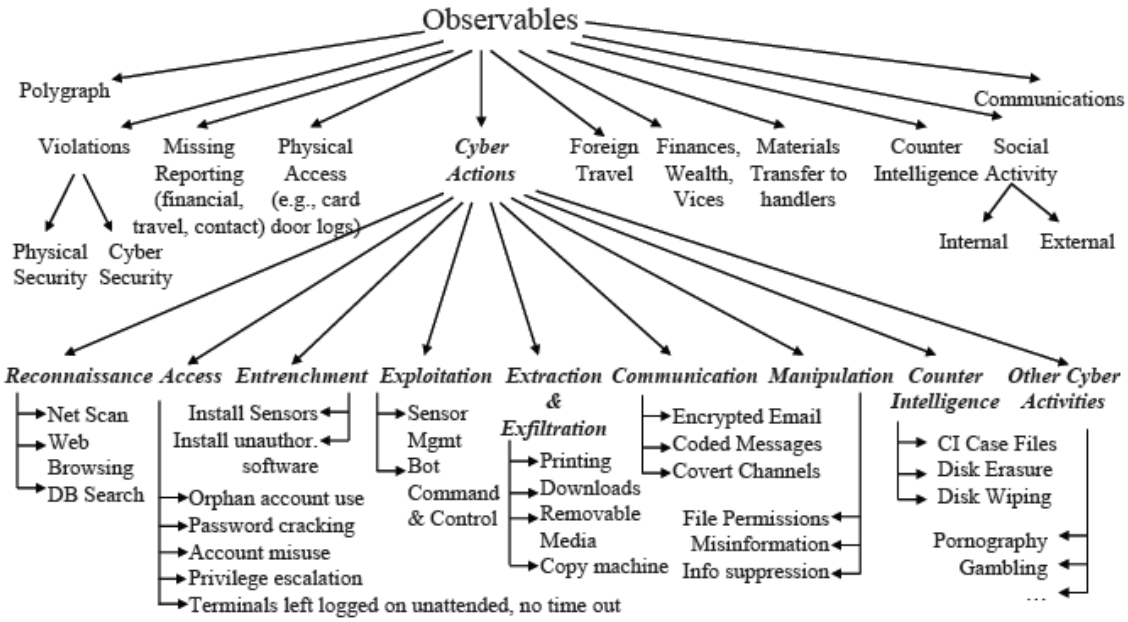

Figure 2.3: Taxonomy of Observables [18]

2.4.3.1. Individual Accountability

Managers can use auditing as a technical measure to help maintain individual accountability. They can help promote good user behavior by ensuring that users know they are personally accountable for their actions and that their actions are tracked by audit logs. Users likely will not attempt to circumvent security policies if they know that their actions are recorded in audit logs, which are subject to periodic review. Audit trails can be used in addition to logical access controls to restrict the use of system resources, to analyze user activities and ensure they have not misused their authorized access or attempted to gain unauthorized access. For example, if an individual has access to sensitive data, such as plans to a new aircraft, audit logs could reveal that those plans had been accessed and printed extensively for weeks before quitting, indicating that the individual misused his/her authorized access to commit a malicious act [15]. As indicated in the RAND study, the malicious insider normally follows a deliberate decision making process, which is presented in the spiral model flowchart in Figure 2.4. Once the malicious insider reaches the risk analysis step, in which he/she assesses the risk of detection prior to committing the attack, he/she then decides whether to continue to deliver the attack. If the individual knows that the risk of detection is high, there is a greater chance that he/she might decide to stop the attack. This is how auditing, if made known throughout the organization can serve as a deterrent to malicious insider activity.

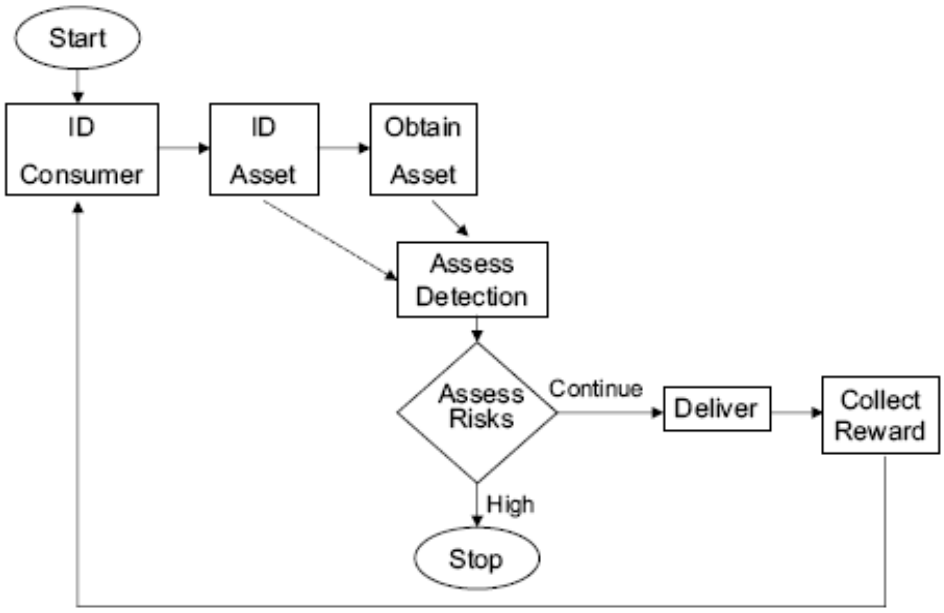

Figure 2.4: Spiral Model Flowchart [18]

2.4.3.2. Reconstruction of Events

Another benefit of auditing is the ability to reconstruct an event after detecting unusual activity, routine errors, or abnormal insider activity, which could present a substantial threat to the organization. A forensic team could also be used to examine audit trails in order to reconstruct events to assist in solving problems [15]. Reviewing audit trails can help determine the event source and the cause of the event, which in turn allows for an easier assessment of the damage. The audit trail can also point to whether the situation was user induced or system created, and possible reasons that it occurred. A good knowledge of the conditions that existed at the time of an event, as well as determining the preexisting conditions that led to the event can aid in avoiding its occurrence in the future.

2.4.3.3. Intrusion Detection

The role of security auditing in an enterprise network can be extremely important because it may often be the only indication of a security breach. If configured to record the appropriate information, an organization can use audit logs to assist in intrusion detection. Intrusion detection is normally considered a real time solution. Although audit logs are not typically considered a real-time solution, advanced auditing can examine audit records as they are created, and generate an alert of the potential intrusion. Many host-based intrusion detection systems work in this manner. After-the-fact, intrusions can be analyzed in depth by using audit logs to determine what happened, and when it occurred, and where the intrusion originated from to ascertain the damage caused by the intrusion [15]. The 2005 CSI/FBI survey found that once an insider attack was detected, system logs were the most prevalent method of identifying the malicious insider. In 76% of these cases, even though the insiders took steps to conceal their identities, the insider was identified using several types of logs as can be seen in Figure 2.5 [7].

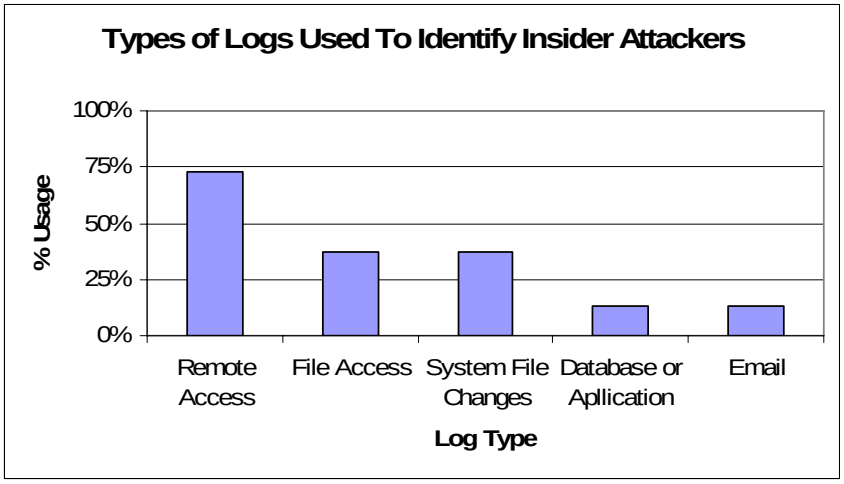

Figure 2.5: Types of Logs Used to Identify Insider Attacks [6]

A disturbing trend in the annual surveys was that even though over half of the respondents surveyed held the position of Chief Information Officer, Chief Security Officer, Chief Information Security Officer, Security Officer, or System Administrator (Figure 2.6), an average of 37% of them did not know if their company had experienced computer crime incidents from the inside (Figure 2.7). With a good auditing policy and regular audit reviews, organizations can detect and stop computer crime incidents before significant damage is inflicted.

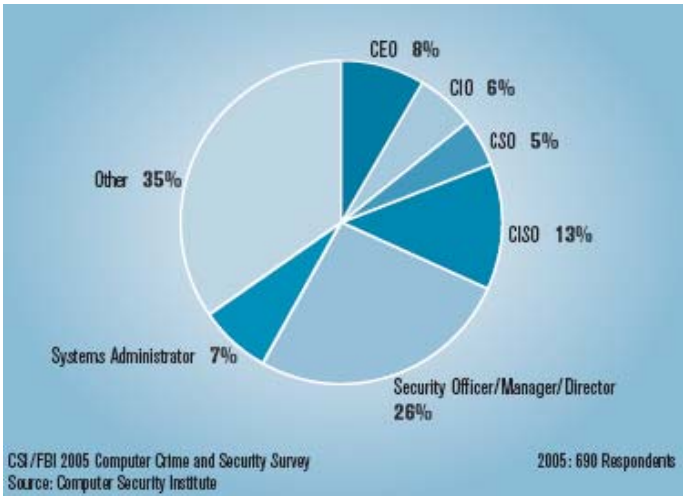

Figure 2.6: 2005 CSI/FBI Respondents by Job Description [7]

How many incidents from the inside, by % of respondents	1–5	6–10	>10	Don't know
2005	46	7	3	44
2004	52	6	8	34
2003	45	11	12	33
2002	42	13	9	35
2001	40	12	7	41
2000	38	16	9	37
1999	37	16	12	35

CSI/FBI 2005 Computer Crime and Security Survey
Source: Computer Security Institute
2005: 453 Respondents

Figure 2.7: Number of Incidents from the Inside [7]

Robert Hanssen, an FBI Supervisory Special Agent was found guilty of what some call “possibly the worst intelligence disaster in US history” [18], when he downloaded large quantities of information from the FBI Automated Case Support System. He also searched the Bureau’s system for any information on his detection, installed unauthorized software on his office workstation, and even hacked onto a Bureau colleague’s workstation. After the fact, audit logs and trails were used to trace his activity and the damage he caused. If the FBI had monitored these audit logs regularly, or in *real time*, they likely would have caught Mr. Hanssen sooner and minimized any damage he caused [18]. After he was caught, he was quoted as saying, “If I thought the risk of detection was very great, I would have never done it.” As a result of his case, the Department of Justice reported the following in its after action report [19]:

> The FBI should implement measures to improve computer security, including:
>
> > (a) Establishing an audit program to detect and give notice of unauthorized access to sensitive cases on a “real-time” basis
> > (b) Establishing an audit program designed to track when employees or contractors are using the FBI’s computer systems to determine whether they are under investigation…

In piecing together the chain of events of an intrusion to determine what occurred in an after-the-fact analysis, audit logs play a major role. The Microsoft Security Risk Management Guide describes a six step approach to follow when investigating incidents. Figure 2.8 provides a graphical representation of the six step process. Auditing plays a key role in the fourth step called “determine cause” which is identifying the cause of damage to a network or individual workstation. A thorough review of the audit logs on

the affected system, as well as other influential devices around them, could give valuable insight into where the attack originated [20].

2.4.3.4. Problem Analysis

Auditing can be used for problem analysis in quasi real-time to identify potential problems in system performance. Quasi real-time auditing of systems can monitor performance status and allow timely reaction to possible critical changes or system halts due to operator input errors or system errors. These indicators pinpoint the cause of errors and allow the organization either to remedy the situation before experiencing

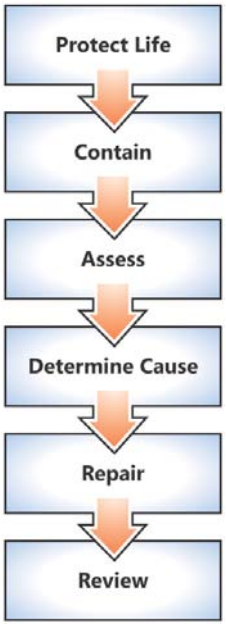

Figure 2.8: Incident Response Process [20]

adverse performance, or to find the root of the problem and solve it. For example, quasi real-time auditing can alert a system administrator when a computer hard drive is filling up unusually fast. This early warning gives the system administrator an opportunity to

investigate the cause of the problem and potentially solve it before the system crashes. A thorough investigation can provide information to prevent future occurrences.

2.5. Related Research

One of the largest areas of research in the area of detecting insider threats has been in the area of intrusion detection. Detection techniques have largely focused in the area of alert correlation analysis. Because of the high amounts of alerts generated by intrusion detection devices and audit logs, various methods have been developed to correlate audit logs and Intrusion Detection System logs to make them easier to manage by security professionals. One area of alert correlation is with the probabilistic approach in which alerts that match closely are fused together in groups of similar events [32, 33]. By matching similarities in these events and then putting them into a hierarchy of similarities, the number of alerts are reduced to a level that is more manageable [34]. Another area of alert correlation is rule based alert correlation, also known as the data-mining approach [23, 35, 36]. In this method, once the alerts are correlated they are scoured for sets of rules based on known attacks. A particular known attack is broken down into steps, identifiable by sets of rules. This approach greatly reduces the number of false alarms, but only works for misuse detection and is not effective for anomaly detection [32].

These research undertakings seem useful in addressing the problems caused by large numbers of intrusion detections being done on large enterprise networks. They address ways of automating the task of sorting through them for indicators of attacks. A particular area that doesn’t appear to have been researched recently is in the area of log

consistency. Specifically, what can be done within an organization using existing workstation auditing, in conjunction with other available technologies, to identify potential insider threats? Research on the configuration of auditing on individual workstation to better identify insider threats is needed so that it can be known that the logs being correlated and scoured are filled with appropriate audit events. That way it can be known that they can detect possible attacks or malicious behavior. This is what this thesis discusses.

2.6. Types and Categories of Auditing

In order to understand the methodology presented in this thesis, the current types of auditing being used, the auditing categories, and how they function within a Microsoft Windows XP® operating system must first be understood.

2.6.1. Workstation Auditing Overview

An organization's audit policy determines the security events to record in order to capture enough details about user, administrator, or system activity. Administrators can monitor security-related activity, such as when a subject accesses an object, when users or administrators log on or off workstations, or if a user attempts to change a system's security policy. The audited events are instrumental in identifying an insider event. Failing to collect key activities hinders the ability to identify potentially malicious insiders. On the other hand, capturing too many activities fills the audit logs with less valuable entries and complicates the filtering process in addition to escalating the cost. Therefore, an organization must plan for a balance between capturing all events and the

minimum required number of activities. In addition, the organization must decide which types of audit logs to monitor and which categories to audit within those logs [21].

Figure 2.9 shows the alert stream complexities phenomenon in which the number of events captured is directly proportional to a larger event space, which induces cost in both storage space and difficulty in filtering appropriate events [22]. This problem is known as alert flooding, in which a large number of alerts are presented to the operator, who has difficulty coping with the load [23]. This phenomenon also occurs with Intrusion Detection Systems.

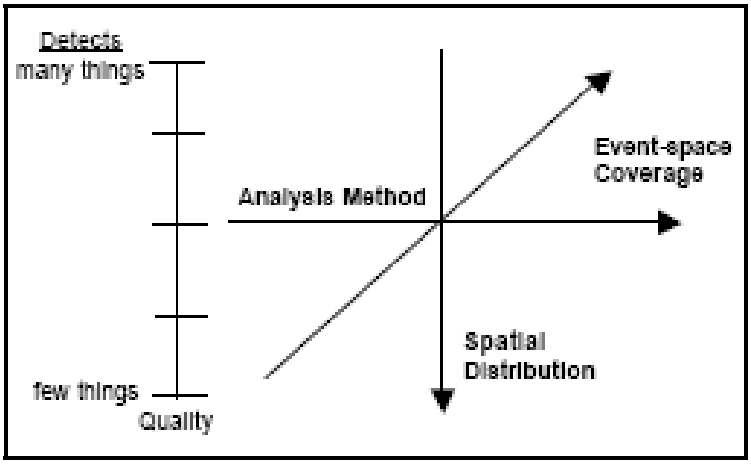

Figure 2.9: Alert Stream Complexities [22]

Maintaining audit logs is a complicated undertaking. With so many users, administrators, and systems being audited the amounts and sizes of the logs can be overwhelming. Since accurately monitoring administrator activities is extremely important and administrators have the capability of manipulating the logs to possibly cover their tracks, it is a good idea to log all activity to a remote logging server. A remote logging server also makes maintaining the audit logs easier.

2.6.2. Remote Logging

With auditing enabled on workstation computers, the default configuration is to log all audited events to the local machine. In a Windows XP® environment, it is logged to the System32/Configuration directory as an ".evt" file. An inside attacker with sufficient skills can delete these files to cover his tracks, so a good practice is to set up remote security auditing, or logging to a remote computer on the network. A Syslog Server approach accomplishes this by the use of various third party software such as NTsyslog or Kiwi Syslog Daemon. The logs stored on a Syslog server should be protected from access by anyone except security personnel [24]. Another advantage of using a remote logging server is that it makes regular automated or manual reviews of these logs less complicated. Also, the storage and backup of these audit logs is easier with the use of a robust server with a lot of storage space and a scheduled backup policy.

2.6.3. Types of Audit Logs

Within most of the Microsoft Windows® operating systems there are three main types of audit logs. They are the application log, the system log, and the security log. Table 2.2 provides a description of each type. On Windows XP® systems the default configuration disables all security auditing. Security administrators must therefore carefully select which categories of events and what types of auditing they want to audit on their organizations workstations [25].

Table 2.2: Types of Audit Logs [26]

Log Type	Description
Application	The application log contains events logged by commercial off-the-shelf applications, and user programs. For example, a database program might record a file error in the application log. Program developers generally decide which events to monitor in this log.
System	The system log contains events logged by system components such as system processes and device drivers. For example, a device driver failure, hardware failure, or failure of a system component to load during startup(such as a service) is recorded in the system log.
Security	The security log can record security events such as changes in user's privileges, changes in the audit policy, file and directory access, and system logons and logoffs. For example, when logon auditing is enabled, an event is recorded in the security log each time a user attempts to log onto the computer.

System and application logging are enabled by the operating system by default. They are both very useful for troubleshooting purposes, but are seldom used in identifying malicious acts. Security auditing captures security events such as unauthorized access, malicious attacks, or unusual occurrences that might warrant investigation [24]. Modification of the local policy to disable or limit security logging requires administrator privileges. By default, any user on a system can view the application and system logs, but only administrators can view the security log. This log, if configured to audit the correct events, is the most useful log to gather information about activity during an incident response. There are nine categories of security auditing, each with separate characteristics and the ability to capture different types of events. Each of these categories can be set to either: No auditing, Success Auditing, Failure Auditing, or both Success and Failure Auditing [27]. Table 2.3 explains each in more detail.

Table 2.3: Auditing Settings [26]

Setting	Description
No Auditing	No audit entry is generated for the associated action.
Success Auditing	An audit entry is generated when the requested action succeeds. For example, a user attempts to log in and is successful it will be logged.
Failure Auditing	An audit entry is generated when the requested action fails. For example, a user attempts to log in and is unsuccessful it will be logged.
Success and Failure Auditing	An audit entry is generated either when a requested action succeeds or fails. For example, a user attempts to login and either succeeds or fails, it will be logged either way.

2.6.4. Categories of Audit Logs

In most Microsoft Operating Systems, specifically Microsoft Windows XP® Professional, there are nine categories of auditable events configurable for auditing. Table 2.4 briefly explains each category. The following sections describes the uses of these configurations as well as an overview of implications of their settings.

2.6.4.1. Audit Account Logon Events

This category is designed for the auditing of remote logons. In this category, events for logging on or logging off remotely are logged by the computer that is used to validate the account. An example of this is a domain controller logging all logons to workstations or shared resources such as file servers or shared printers within its domain. For a local machine account, the local machine logs local logins into this category for a local account.

2.6.4.2. Audit Account Management

In this category, events are logged tracking attempts to create new users or groups, rename users or groups, enable or disable user accounts, change account

Table 2.4: Auditing Categories [28]

Category	Effect
Audit account logon events	Audits logon attempts to a local account on a computer. If the user account is a domain account, this event also appears on the domain controller.
Audit account management	Audits the creation, modification, and deletion of user and group accounts, in conjunction with password changes and resets.
Audit directory service access	Audits access to objects in the Active Directory service.
Audit logon events	Audits attempts to log on to workstations and member servers.
Audit object access	Audits attempts to access an object such as a file, folder, registry key, or printer that has defined audit settings within that object's system access control list (*SACL*).
Audit policy change	Audits any change to a user rights assignment, audit, account, or trust policies.
Audit privilege use	Audits each instance that a user exercises a user right, such as changing the system time.
Audit process tracking	Audits application behavior such as program starts or terminations.
Audit system events	Audits computer system events such as startup and shutdown and events that affect system security or the security log.

passwords, and enable auditing for account management events. If this category is enabled for auditing, administrators can track events to detect malicious, accidental, and authorized creation of user and group accounts. This category is best suited for keeping an eye on the actions of administrators to make sure they are performing within the parameters of the organization security policy.

2.6.4.3. Audit Logon Events

When auditing "logon events", the system records events involving the creation and destruction of logon sessions to its local machine regardless of whether the local

machine is on the network or not. These events occur on the computer accessed, as opposed to the one that validates the account and may not necessarily be the one accessed as in Audit Account Logon Events. For example, if a network logon was performed to access a share on a file server, these events are generated on the file server itself. Without selecting this category for auditing, it is difficult or impossible to determine which user has either accessed or attempted to access computers in an organization. The events in this category are useful for detecting a malicious user unplugging a workstation from the network, breaking into it, then attempting to gain access to the network.

2.6.4.4. Audit Object Access

Enabling object access auditing records the events of a user accessing an object, such as a file, folder, registry key, or printer that has a specified System Access Control List (*SACL*) configured. By itself this policy setting does not log any events, because each individual object audited must have a *SACL* configured. A *SACL* is comprised of Access Control Entries (*ACE*). Each *ACE* contains three pieces of information [21]:

- The security principal (user, computer, or group) to be audited when they attempt to access the object.
- The specific access type to be audited (such as read data, write data, delete, or full control), called an access mask.
- A flag to indicate whether to audit failed access events, successful access events, or both.

Organizations should carefully consider the actions to capture when they configure *SACL*s, because they can be extremely useful in detecting particular insider threats based on security goals. For example, it is a good practice to enable tracking of

user writes and appends on executable files to track when they are changed or replaced, because computer viruses, worms, and Trojan horses typically target executable files. Additionally, tracking user accesses and modifications of sensitive documents can help to determine possible unauthorized access or malicious activity [29].

2.6.4.5. Audit Policy Change

This category configures auditing of every incident of a change to user's rights assignment policies, trust policies, or changes to the audit policy itself. This category assists in monitoring the activities of administrators. Any configuring or changing of any of the workstation or server policies within an organization should be investigated to determine if it was authorized or not. This category identifies malicious users attempt to circumvent auditing by changing the audit policy or clearing the audit logs, which is a common technique used to cover up any malicious insider activity.

2.6.4.6. Audit Privilege Use

This category configures auditing of each instance of a user exercising a user right. Examples of this are a user exercising the shutdown system privilege to shut down or reboot a machine, or the take ownership privilege of a file or directory. These events can be informative about the occurrences of malicious activity but can generate a very large number of event records.

2.6.4.7. Audit Process Tracking

This category configures auditing of detailed tracking information for events such as program activation, process exit, handle duplication, and indirect object access. If the organization security policy lists authorized processes executable by users and

administrators within the organization it can identify unauthorized programs running such as a Trojan horse, a rootkit, or other type of malicious software. It can also be beneficial during an incident response by providing a detailed log of the processes started and the time when they were started [21].

2.6.4.8. Audit System Events

This category audits system events such as starting or shutting down computers, event logs filling up, the audit logs being cleared, or other security-related events that affect the entire system. These events may help determine instances of unauthorized system access, or critical circumstances that affect system security. For example, clearing the audit logs would log an event 517, which would indicate that the audit log was cleared. This could be an indication of a malicious user covering his/her tracks.

2.6.5. Security Templates

By default, Windows XP® is designed with user ability in mind and has limited security enabled. Even fully patched with Service Pack 2 and all security updates, Microsoft does not recommend the use of Windows XP® *out-of-the-box* default security configuration on most enterprise networks [21]. There are many security settings that can be configured, and Microsoft has suggested settings for an enterprise network. In order to ensure that enterprise network security policies are applied to enterprise workstations, security policies on individual workstations can be deployed with the use of security templates. Security templates are text based files that contain values for security related system settings, and make configuring those settings easier to manage and implement. Administrators can create and update them using the Security Templates Microsoft

Management Console (*MMC*) snap-in. These templates can be applied to a local computer as an individual file, or to an entire group using Active Directory.

To configure a Windows XP® workstation for enterprise client network use, Microsoft suggests starting with their enterprise client security template and configure it for the organizations needs [21]. There are other recommended security templates available such as those recommended by the National Institute of Standards and Technology (*NIST*) and the National Security Administration (*NSA*).

These security templates are designed for desktop workstations on an enterprise network. Among these are the *NIST* Enterprise Client Desktop (*NIST EC*), *NIST* Specialized Security Limited Functionality Desktop (*NIST SSLF*), and *NSA* Systems and Network Attack Center Enterprise Client Desktop (*NSA SNAC*). Table 2.5 contains a comparison between the audit settings of these configurations.

2.6.5.1. *NIST EC, NIST SSLF*, and *NSA SNAC EC* Security Templates

The *NIST* as well as the *NSA* have worked together with the Defense Information Systems Agency (*DISA*), the Federal Bureau of Investigation (*FBI*), SysAdmin Audit Network Security Institute (*SANS*), Center for Internet Security (*CIS*) and other vendors to develop a set of benchmark security guides to provide a basic security template of security settings for use in various environments [31]. *NIST* has developed several security templates for Windows XP® that are recommended for use within enterprise networks. For typical enterprise network use they recommend using the Enterprise Client security template. The enterprise environment, also known as managed environment,

typically consists of large organizational systems with defined suites of hardware and software configurations. They usually consist of centrally managed workstations and servers protected from threats on the internet with firewalls and other network security devices [30]. For systems with a higher risk profile or risk exposure, *NIST* recommends the Specialized Security-Limited Functionality (*SSLF*) security template. These include systems that contain confidential information (i.e. personnel records, medical records, and financial information) or perform vital organizational functions (i.e. accounting, payroll processing, air traffic control) [30]. These systems are more likely to be attacked

Table 2.5: Audit Settings of Security Templates[21, 30, 31]

Audit Setting	Windows XP® Out of the Box	*NIST EC* Template	*NIST SSLF* Template	*NSA SNAC* Template
Audit account logon events	No Auditing	Success	Success, Failure	Success, Failure
Audit account management	No Auditing	Success	Success, Failure	Success, Failure
Audit directory service access	No Auditing	No Auditing	No Auditing	No Auditing
Audit logon events	No Auditing	Success	Success, Failure	Success, Failure
Audit object access	No Auditing	No Auditing	Failure	Failure
Audit policy change	No Auditing	Success	Success	Success, Failure
Audit privilege use	No Auditing	No Auditing	Failure	Failure
Audit process tracking	No Auditing	No Auditing	No Auditing	No Auditing
Audit system events	No Auditing	Success	Success	Success, Failure

by either outside or inside malicious parties. The auditing security settings in the *NIST SSLF* security template are considerably more intensive than in *NIST EC*, and much more information can be obtained from the audit logs. The drawback is that there is much more information logged, resulting in a lot of information to sort through in order to find useful information about an incident. The *NSA SNAC* Security Template is similar to the *NIST SSLF*, but has some minor differences in the auditing configuration. Table 2.5 shows the differences between the auditing settings of the security templates. In this thesis, the auditing capabilities of these security templates will be tested and compared with a customized auditing template created by the presented methodology.

2.7. Summary

This chapter has described the objectives for maintaining information security within an enterprise network. It has also covered trends, characteristics, and types of insider threats. Attack methods and ways of identifying insider threats were also presented. Next, auditing was covered with a discussion on the definition of auditing, the audit policy in an organization, and the role of auditing in the enterprise. Finally the types of auditing, the categories of auditing, and the use of security templates were discussed.

III. Methodology

3.1. Chapter Overview

This chapter outlines the methodology used for developing a customized auditing template for a computer workstation, the calculations and metrics used in this process, and a method for evaluating the end product. It begins with a presentation of the research goals and the approach taken to achieve these goals. Following sections define the system under test, the component under test, and the experimental design. Next the system setup and data collection are presented. Finally, the evaluation technique, calculations performed, and metrics used are described.

3.2. Problem Definition

In addition to perimeter defenses, organizations often take advantage of available resources to ensure security within the organization, such as the existing auditing capabilities of Windows XP®. Often this is done blindly using existing security templates recommended for use by *NIST* and *NSA*, without research into the auditing configurations these templates contain, and how they meet their organizations requirements for computer security. Rather than rely on existing client workstation auditing configurations, an organization should know exactly what events its client workstations are auditing, and configure the clients to audit based on computer security requirements of the organization [29]. Currently there exists no defined methodology for creating a customized auditing template based on organizational security requirements.

3.2.1. Research Goals

The primary goal of this research is the establishment of a methodology for developing a customized auditing template for a computer workstation to audit for insider threats. It develops an insider threat client workstation-auditing model to create a tailored auditing template for a Windows XP® workstation based on organizational computer security requirements. This model uses the existing auditing capabilities of Windows XP® and fine-tunes them to audit malicious insider activities.

3.2.2. Approach

An evaluation of the auditing categories available for configuration in Windows XP® is performed to determine which configuration best detects the selected simulated malicious insider scenarios. The categories available for configuration within Windows XP® will each be enabled individually, one at a time, while all others are disabled to evaluate the events logged for a particular category. There are currently nine auditing categories in Windows XP®. A series of 18 malicious insider scenarios, as well as non-malicious scenarios consisting of simulated typical user behaviors, are simulated and the logs resulting from these activities are analyzed. There are 18 scenarios simulated, and 15 minutes of activities representative of normal user activity. The two questions to be answered are:

1. Was the malicious activity detected in the logs with the current category enabled? This would indicate that the particular category is either useful or not for detecting that type of activity.
2. What is the particular cost involved in this identification? In other words, what is the ratio of the events that identify the malicious activity

to the number of other events in the logs? This will indicate the ease of identifying the malicious activity.

These two questions are answered for each particular category of auditing within Windows XP®. All categories are then compared, and selected or deselected for auditing based on the criteria presented later in this chapter. The end result is a customized auditing template based on the malicious scenarios selected. Finally, an evaluation of the auditing template can be made based upon other preexisting auditing configurations inside one-size-fits-all security templates (e.g. *NIST EC*, *NIST SSLF*, *NSA SNAC*, etc.)

3.3. System Boundaries

The system boundaries consist of the System Under Test (SUT), the Component Under Study (CUS), the type of data input into them, and the processing of the output from them.

3.3.1. Auditing System

The System Under Test (*SUT*) is the Windows XP® auditing system running inside a virtual computer, using VmWare Workstation (Figure 3.1). It is a basic installation of Windows XP® Professional with Service Pack 2 and all Microsoft security patches as of February 28, 2006. The SUT includes an adjustable auditing configuration, which is the Component Under Study (CUS). The parameters are the nine auditing categories, configured one at a time to audit success and failure, while all other categories record nothing. The input for the SUT is the group of 18 simulated malicious scenarios, and the non-malicious (normal user) scenarios. All scenarios are manually executed through the SUT, independently one at a time while the CUS audits for success and

failure. The Windows XP® system audits accordingly and records detectable events in a security log. After each test run is complete, the virtual machine is reset to the baseline configuration. This is accomplished using a VmWare snapshot.

The scope of the experiment is limited to a single workstation operating in VmWare without network connectivity. It is designed to simulate the security implications of multiple users sharing a common workstation. The workstation does not include the logging of network access and interaction with other workstations, servers, or peripheral devices.

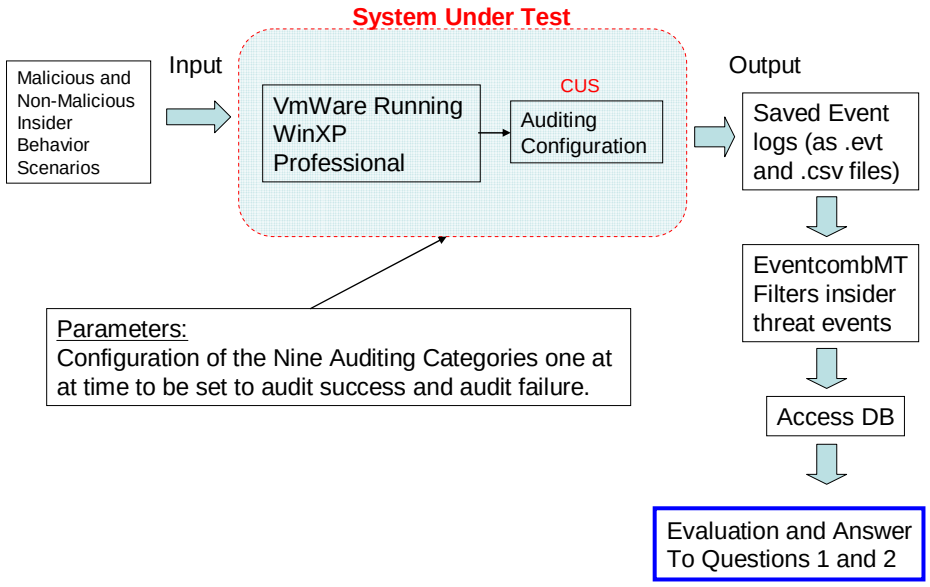

Figure 3.10: Windows XP® Auditing System

3.4. Parameters

The system parameters are the audit category settings. Currently there are nine audit categories for the Windows XP® workstation. Each category has four configuration settings, (1) no auditing, (2) audit success, (3) audit failure, or (4) both audit success and

failure. To represent these, each of the nine categories is represented by 2 switches, one for success audit, and one for failure audit. This represents each of the four configurations; (1) Neither success switch nor failure switch is enabled = no audit, (2) success switch is on, failure switch is off = success audit, (3) success switch is off, failure switch is on = failure audit, and (4) both success and failure switches are on = both success and failure audit. Table 3.1 shows these switches and their corresponding audit categories. The resulting logs are parsed into success and failure audits for further evaluation and analysis.

Table 3.1: Switches and Corresponding Categories and Settings

Switch	Corresponding Category and Setting
S1	Success Audit Account Logon Events
F1	Failure Audit Account Logon Events
S2	Success Audit Account Management
F2	Failure Audit Account Management
S3	Success Audit Directory Service Access
F3	Failure Audit Directory Service Access
S4	Success Audit Logon Events
F4	Failure Audit Logon Events
S5	Success Audit Object Access
F5	Failure Audit Object Access
S6	Success Audit Policy Change
F6	Failure Audit Policy Change
S7	Success Audit Privilege Use
F7	Failure Audit Privilege Use
S8	Success Audit Process Tracking
F8	Failure Audit Process Tracking
S9	Success Audit System Events
F9	Failure Audit System Events

3.5. Data Collection

The output from the SUT for each scenario is parsed into success and failure fields in a Microsoft Excel® worksheet, and then grouped by category. EventcombMT, a free utility available from Microsoft, is used to parse the logs to find known logged

events that correspond to the malicious actions performed. EventcombMT is configured with a list of common events which are indicative of malicious insider activities. The EventcombMT output is saved to a Microsoft® Access Database for further analysis to identify simulated malicious insider actions.

3.6. System Setup

The virtual machine consists of VmWare, version 5.0.0, build 13124. The hardware platform consists of a Toshiba laptop with a Pentium IV 3.06 GHz Processor, and 896 MB of Random Access Memory. The Virtual Operating System is Windows XP® Professional. The operating system is patched with Service Pack 2 and all available security patches from the Microsoft Update website as of 28 February 2006.

3.7. Procedure

Each time a group of scenarios is simulated, the following steps are completed to ensure that the samples obtained are completely individual and specific to the current switch setting:

1. Load VMWare snapshot – This resets the virtual machine to the baseline for each group of scenarios to run within a specific audit category.
2. Configure Category – The particular audit category under test is configured to enable both success and failure auditing. All others are set to no auditing.
3. Wipe audit logs – All logs (application, security, and system) are cleared to ensure that there are not any other logged events in them.
4. Perform test run – The 18 malicious scenarios, as well as the non-malicious scenarios (simulated normal user activity) are manually executed using a checklist of exactly what is to be performed.

5. Save log files – Upon completion of the test run, all audit logs (application, security, and system) are saved as .evt and .csv files for later analysis offline.
6. Evaluate log files – Upon completion of all scenarios for a particular audit category, the security audit logs are evaluated to determine if they logged the identifying events, and their events are separated by scenario for further analysis.

3.8. Experimental Design

The experimental design for this research is a factorial design with one factor, the auditing settings [37]. The source of randomness in this experiment was the scenarios.

The number of experiments performed:

- Number of scenarios = 18
- Number of auditing settings = 9 categories * 4 settings each = 36

Total number of experiments (18) * (36) = 648

3.9. Malicious Scenarios

A critical factor in the methodology is the selection of malicious scenarios. An organization needs to select these scenarios carefully to ensure they represent their identified insider threat security requirements. For this experiment, the scenarios are selected based upon possible requirements of a typical organization. For this experiment eighteen scenarios are selected as an example of possible scenarios an organization might select. The scenarios focus on protecting the confidentiality of files on a shared workstation. The scenarios simulate multiple users using a common workstation, and a malicious user attempting to access other users files. The scenarios simulate various techniques that a malicious insider might use to obtain access to vital information.

Privileges held by administrators as well as normal users are represented in the scenarios. The scenarios are presented in Appendix A and described step by step in Appendix B.

3.10. Non-malicious Scenarios (Simulated Normal User Activity)

In order to gain an understanding of the size and typical number of entries captured in a workstation audit log, some non-malicious, *normal user*, activities are simulated. This consists of typical activities that normal users do in day-to-day use of a workstation. A selection of 15 minutes worth of typical normal activities such as word processing, spreadsheet creation, and internet browsing were used in the experiment. A list of these activities can be found in Appendix C. The events logged are then extrapolated to get one weeks worth of normal user activity. To simulate this, the 15 minutes of normal user activity is multiplied by 160. This one-week worth of normal user activity is called the Calculated Normal User Count (*CNUC*). It is used in calculations to help determine the best auditing configuration for logging the malicious scenarios. One week of normal user activity is chosen as an example, and an assumption is made that a security administrator can check a particular workstation once a week for malicious activity and would therefore sort through one week's worth of normal user activity. In a real case situation, an organization would know its security administration procedures and select a period of time more realistic to their organization based on their security procedures.

The non-malicious activity is not intended to be directly representative of typical normal user activity, but only to serve as a controlled, consistent method of generating typical user activities in no specific order. This strategy helps introduce deterministic

activity with malicious activity to the *SUT*. Another approach is for an organization using this model to generate an average event count from actual logs on various workstations within the organization, which would be a more accurate representation of normal user behavior.

3.11. Evaluation Technique

Once the results from the malicious scenarios are collected and evaluated, they are put into a table for ease of analysis to determine the best selection of switches for an organization based on their security requirements, an example is shown in Table 3.3. In this example, the scenario detections are in bold. This table, along with the calculations presented in the section 3.10.1, and the selection criteria presented in section 3.10.2 will enable an organization to select the best categories to select for auditing in their audit configuration.

Table 3.3: Breakdown of Events and Detections

	Categories (Switches)																	
Scenarios	**S1**	**F1**	**S2**	**F2**	**S3**	**F3**	**S4**	**F4**	**S5**	**F5**	**S6**	**F6**	**S7**	**F7**	**S8**	**F8**	**S9**	**F9**	**Scenario Detected**
1a	**1**	0	**1**	0	0	0	**1**	0	**1**	0	**1**	0	**4**	0	**2**	0	**2**	0	**Yes**
2a	0	0	0	0	0	0	0	0	**15**	0	0	0	0	0	2	0	0	0	**Yes**
3a	0	0	0	0	0	0	0	0	**62**	0	0	0	0	0	2	0	0	0	**Yes**
4a	0	0	0	0	0	0	0	0	0	**108**	0	0	0	0	0	0	0	0	**Yes**
5a	1	7	**2**	0	0	0	6	6	86	0	0	0	1	0	35	0	0	0	**Yes**
6a	3	7	**6**	0	0	0	11	7	21	0	0	0	11	0	88	0	0	0	**Yes**
7a	2	8	**1**	0	0	0	7	8	33	0	0	0	10	0	47	0	0	0	**Yes**
8a	0	1	**1**	0	0	0	9	2	17	0	0	0	10	0	46	0	0	0	**Yes**
9a	3	2	0	0	0	0	2	1	**6**	0	0	0	5	0	12	0	0	0	**Yes**
10a	0	1	0	0	0	0	2	0	0	**1**	0	0	0	0	12	0	0	0	**Yes**
11u	1	0	0	0	0	0	2	1	0	0	0	0	1	0	10	0	0	0	**No**
12u	0	0	0	0	0	0	0	0	**54**	0	0	0	0	0	3	0	0	0	**Yes**
13u	0	0	0	0	0	0	2	0	0	**108**	0	0	0	0	9	0	0	0	**Yes**
14a	0	1	0	0	0	0	4	1	**76**	78	0	0	13	0	22	0	0	0	**Yes**
15a	**2**	1	**1**	0	0	0	**3**	1	**5**	0	**1**	0	**6**	0	**13**	0	**1**	0	**Yes**
16u	**1**	4	0	0	0	0	**9**	**4**	12	3	1	0	39	0	40	0	28	0	**Yes**
17u	2	0	0	0	0	0	35	0	**73**	0	12	0	22	0	36	7	**16**	0	**Yes**
18u	1	1	0	0	0	0	2	1	0	0	11	0	8	0	**30**	0	0	0	**Yes**
Total Events	**17**	**33**	**12**	**0**	**0**	**0**	**95**	**32**	**461**	**298**	**26**	**0**	**130**	**0**	**409**	**7**	**47**	**0**	
Total Detects Per Switch	**3**	**0**	**6**	**0**	**0**	**0**	**3**	**0**	**7**	**3**	**2**	**0**	**2**	**0**	**3**	**0**	**3**	**0**	

3.11.1. Auditing Template Creation Methods

There are two methods used for creating a custom auditing template, the Switch Evaluation Method (*SEM*), and the Configuration Evaluation Method (*CEM*). Each serves a different purpose, and it is suggested that an organization try both methods to determine which works best for their security requirements.

3.11.2. Switch Evaluation Method

The *SEM* uses calculations, metrics, and organization established criteria to assist in creating a Windows XP® auditing template based on a per switch evaluation. It uses metrics for evaluating the quality of a given switch at detecting the malicious act, as well as the cost involved in that detection. The development of metrics allows the comparison of the switches to determine the best combination to put into the Windows XP® workstation auditing template. In this method, seven metrics were created for measuring the effectiveness of a given auditing category given that a set of malicious scenarios is encountered. The metrics used for this method are:

1. Switch Detection Coverage (*SDC*)
2. Calculated Normal User Count (*CNUC*)
3. Switch Activity Cost (*SAC*)
4. Total Activity Cost (*TAC*)
5. Switch False Positive Count (S*FPC*)
6. Switch Figure of Merit, (*SFOM*)

Switch Detection Coverage (*SDC*), is the metric that identifies how well a given switch performs at detecting a given set of scenarios. Calculated Normal User Count

(*CNUC*), is a calculated amount of simulated weekly normal user activities. Switch Activity Cost (*SAC*), is the total number of logged events for a given switch after running the malicious scenarios, the non-malicious scenario, and calculating the *CNUC*. Total Activity Cost (*TAC*), is the total of all switch *SAC*'s. Switch False Positive Count (*SFPC*), is the total number of false positives of a switch after running the malicious scenarios and calculating the *CNUC*. After the previous metrics have been derived, the Switch Figure of Merit (*SFOM*), is created by constructing a weighted function of these metrics. The *SFOM* metric will allow for an objective comparison of performance between the individual switches. The weights assigned in the function are selected to scale the importance of each of the constituent metrics according to organizational security requirements. In addition, formulation of the problem in this form enables the optimal selection of auditing categories by using existing linear optimization methodologies.

In order to represent each of the metrics and the switch figure of merit, the problem is loosely framed in the context of set theory [38]:

Auditing configuration is determined by setting or clearing a set of 18 auditing switches, both success switches and failure switches (. Each auditing switch takes on one of two discrete values, either on or off:

$$Value = \{0,1\} \qquad \text{(Eq. 3.1)}$$

A Switch, S_x where $\underline{x} = [1..M]$, is assigned an element of the set *Value*:

$$S_x = Value \qquad \text{(Eq. 3.2)}$$

An auditing configuration, *AC*, consists of a set of *M* Switches. When an auditing configuration is evaluated in it's entirety based on its current set of *M* switches, it is referred to as AC_X:

$$AC_x = \{S_1, S_2, ..., S_M\} \quad \text{(Eq. 3.3)}$$

The scenarios, *SC*, consist of a malicious scenario set, SC_M, a non-malicious scenario set, SC_N. The malicious scenario set, SC_M, consists of the 18 selected simulated malicious scenarios:

$$SC_M = \{SC_{M_1}, SC_{M_2}, ..., SC_{M_{18}}\} \quad \text{(Eq. 3.4)}$$

The non-malicious scenario set, SC_N, consists of one 15 minute scenario consisting of simulated typical normal user activity:

$$SC_N = \{SC_{N_1}\} \quad \text{(Eq. 3.5)}$$

The events are the events that appear in the security audit log after the scenarios are executed. They consist of the events from the malicious scenarios, E_M, and the events from the non-malicious scenarios, E_N. The total events for a given switch, S_x, after all malicious scenarios, E_M, and non-malicious scenarios, E_N, have been executed are E_x where $(x \mid x \in \{S_1, S_2, ..., S_M\})$. The set of all possible events is the total of all events for all switches, E_T, as $E_T = \bigcup E_x$ where $(x \mid x \in \{S_1, S_2, ..., S_M\})$.

In the following sections, we loosely use set theory notation to define metrics that represent the effectiveness of a given Switch, *S*, and a given overall AuditConfiguration, *AC*, when subjected to a given set of scenarios, SC_M and SC_N.

3.11.2.1. Switch Detection Coverage

One of the most important metrics in evaluating a switches merit is how well it is at detecting a set of scenarios. For this reason, the first metric introduced is the Switch Detection Coverage (*SDC*). A given malicious scenario is deemed to be detected if the critical event appears in the security log file. The *SDC* is a measure of how well an individual audit switch did at finding the set of malicious scenarios, SC_M. The set of malicious scenarios detected, SC_D, where $SC_D \subseteq SC_M$. It is calculated as the number of detected scenarios, SC_D, divided by the total number of malicious scenarios, SC_M, in the malicious scenario set for a given audit switch configuration.

$$SDC\big|_{Sx} = \frac{|SC_D|}{|SC_M|} \qquad \text{(Eq. 3.6)}$$

The *SDC* is an unbiased indicator at how well a particular audit switch, S_x, is at detecting the malicious behavior contained in a given malicious scenario set, SC_M.

For example, suppose an organization, in using the insider threat client workstation-auditing model to create their auditing configuration, has created a set of 15 malicious scenarios in order to determine the effectiveness of the current audit switches. They would run each of the scenarios contained in the scenario set for each switch and determine which of the scenarios are detected. If 10 of the 15 total scenarios are detected for a given switch, the *SDC* calculation for that switch would be:

$$SDC\big|_{S_x} = \frac{|SC_D|}{|SC_M|} = \frac{10}{15} = 0.667 \qquad \text{(Eq. 3.7)}$$

While *SDC* is an effective measure of the detection coverage of a given audit switch under a given set of scenarios, it fails to account for the cost in log file size and false positives encountered.

3.11.2.2. Activity Costs

Activity Costs (*AC*) are considered an obstruction to identifying the events that indicate the malicious act. They are events captured that are not indicative of a malicious act. *AC*'s conceal and obfuscate the events necessary to identify a malicious act. They can be thought of as "background noise". They appear in every malicious and non-malicious scenario and must be filtered out to find the events that indicate a malicious act has occurred.

False Positive Events (*FPE*) have a greater cost than activity costs because they are misleading. *FPE*'s often make the security administrator think that they indicate a malicious act when in fact there is no malicious activity. The extra time it takes for a security administrator to evaluate and determine the legitimacy of an *FPE* can be very costly.

3.11.2.3. Calculated Normal User Count

The auditing switches differ from one another in terms of the events, E_X, they generate in the security log files. Each event in the log file is an indicator of a specific event occurring and incurs a cost in terms of disk storage. To add realism to the model, non-malicious normal user activities are simulated to add to the size and complexity of the security logs. Non-malicious normal user events are the events that are logged continuously, and must be sorted through to find malicious activities. For this study, non-

malicious normal user events, E_N, are calculated for the period of a week, to simulate the log size of a workstation that is examined weekly. To calculate the Calculated Normal User Count, *CNUC*, the 15 minute non-malicious scenario events, E_N, is multiplied times 160 to represent a week.

$$CNUC\big|_{S_x} = 160 \text{ x } |E_N| \quad \text{(Eq. 3.8)}$$

For example, suppose an organization is using the insider threat client workstation-auditing model and has created a set of 15 non-malicious scenarios contained in SC_N in order to determine the effectiveness of a particular audit switch. If they want to determine the *CNUC* obtained using the current audit switch, S_x, they would run the non-malicious scenario set for 15 minutes, sum the number of normal user events that were added to the security log, multiply by 160 to get a weekly count. If they had 17 normal user events added to the security log in 15 minutes, the calculation would be:

$$CNUC\big|_{S_x} = 160 \text{ x } 17 = 2720 \quad \text{(Eq. 3.9)}$$

3.11.2.4. Switch Activity Cost

As noted earlier, each event, or line in a log file, incurs a cost in terms of disk storage and obfuscation. For this reason, activity cost for each switch must be determined. The Switch Activity Cost (*SAC*), is a measure of how many lines are generated in the security log files resulting from a given auditing switch, S_x, both from the malicious scenarios and from the Calculated Normal User Cost (*CNUC*). It is calculated by taking the sum of the malicious scenarios events, E_M, and the *CNUC* (which is E_N times 160):

$$SAC\big|_{S_x} = |E_M| + CNUC\,|\,S_x \quad \text{(Eq. 3.10)}$$

For example, suppose an organization has created a set of 15 malicious scenarios contained in SC_M, and 15 minutes of non-malicious normal user activity contained in SC_N. If they want to determine the *SAC*, of a particular audit switch, S_X, they could test each of the scenarios contained in the scenario set and the 15 minutes of normal user activity and compute the *CNUC* for the switch. They then add the malicious scenario events, E_M, to the *CNUC*. If E_M is 500 events, and the *CNUC* was 2720, the calculation would be:

$$SAC\big|_{S_x} = (500 + 2720) = 3220 \qquad \text{(Eq. 3.11)}$$

While *SAC* is an effective measure of the cost in log file lines associated with a given audit switch under a given set of malicious and non-malicious scenarios, it fails to account for the false positives encountered.

3.11.2.5. Total Activity Cost

The Total Activity Cost, *TAC*, is the total number of activity costs for all switches, S_T, including malicious scenarios, SC_M, and the *CNUC* (calculated from the non-malicious scenarios SC_N). It is the set of all possible events, or the total of all events for all switches, E_T, as $E_T = \bigcup E_x$ where $\left(x \mid x = \{S_1, S_2, ..., S_M\}\right)$. It is calculated as the sum of the *SAC*'s for all switches, or the union of all switch total events, E_X:

$$TAC\big|_{S_T} = \bigcup E_x \qquad \text{(Eq. 3.12)}$$

For example, suppose a given organization has created a set of 15 malicious scenarios contained in SC_M, and calculated its *CNU* from the non-malicious scenario, SC_N, and the *SAC* results are as follows:

Switch	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9
SAC	1320	543	36	1003	920	45	890	63	0	943	54	5	0	0	31	133	920	0

The calculation would be:

$$TAC\big|_{Sx} = \begin{pmatrix} 1320+543+36+1003+920 \\ +45+890+63+0+943+54 \\ +5+0+0+31+133+920+0 \end{pmatrix} = 6906 \qquad \text{(Eq. 3.13)}$$

3.11.2.6. Switch False Positive Count

Since the detection of a specific scenario is dependant on the detection of a single critical event and not a specific chain of events, it is possible that we may believe that we have detected malicious behavior when in reality the critical event resulted from a legitimate system action. For this reason, the next metric introduced is the Switch False Positive Count (*SFPC*). The S*FPC* is a measure of how many critical events occurred that did not correspond to a malicious scenario given an audit switch, S_x , the set of malicious scenarios, SC_M, and the set of non-malicious scenarios, SC_N. It is calculated as the sum of critical events not detecting a malicious scenario that occurred in the log file resulting from the set of malicious scenarios, and the *CNUC*. It is simply a count of events that have the same event numbers as detection but are not detections in the *SAC*. This is not easily defineable by set theory:

For example, suppose a given organization created a set of 15 malicious scenarios contained in SC_M, in the current audit switch, S_x, and there were 3 False Positive Events, and they also executed a set of 15 minutes of non-malicious scenarios contained in SC_N, and calculated the *CNUC* which produced 32 False Positives. The *SFPC* would be:

$$SFPC\big|_{S_x} = 3 + 32 = 35 \qquad \text{(Eq. 3.14)}$$

3.11.2.7. Switch Figure of Merit

The Switch Figure of Merit (*SFOM*) is a weighted metric that applies a value to each switch based on its performance given the selected scenarios. It allows the switches to be objectively compared against each other. To calculate the *SFOM*, the calculations from the following metrics that have been calculated are used. They include the following:

1. Switch Detection Coverage (*SDC*)
2. Calculated Normal User Count (*CNUC*)
3. Switch Activity Costs (*SAC*)
4. Total Activity Costs (*TAC*)
5. Switch False Positive Count (*SFPC*)

Each of the metrics identified above provide valuable information about the quality of an audit switch, S_x, given a set of malicious scenarios, SC_M, and a corresponding set of non-malicious scenarios, SC_N. However, it is desirable to create a single figure of merit that allows for quick and easy comparison of audit categories to allow the optimal selection of an auditing template. For this reason, the Switch Figure of Merit (S*FOM*) metric is introduced, combining the Switch Detection Coverage (*SDC*), the Switch Activity Costs (*SAC*), the Total Activity Costs (*TAC*), and the Switch False Positive Count (*SFPC*) into a single, derived metric:

$$SFOM\big|_{S_x} = W_1 \text{ x } SDC\big|_{S_x} - W_2 \text{ x } \left[\frac{SAC}{TAC}\right]\Big|_{S_x} - W_3 \text{ x } \left[\frac{SFPC}{SAC}\right]\Big|_{S_x} \qquad \text{(Eq. 3.15)}$$

No similar metric could be found for this type of application, so this metric was contrived specifically for this methodology. The metric takes into consideration, the 3 most important components of an audit category; the detections, the activity costs, and

the false positives. There exists a balance between these three components. The positive component (detections) is offset by the negative components (activity costs and false positives). If a large number of detections are desired, and auditing switches are selected accordingly to audit more events, then the negative affect is an increase in activity costs and false positives. On the other hand, if selecting audit switches is based on attempting to lower activity costs or false positives, then the total detections will be lowered. This metric takes these factors into consideration and provides the capability to assign a weight to each of the components of the metric based upon their importance to the organizational mission. In parallel to Statistical Decision Theory, in which statistics are integrated with decision-making, the selection of weights based upon organizational priorities assists in the organization's creation of an auditing template by enabling a tailored organizational balance between high detection rates, log file size, and false alarms [39]. The higher each weight, the more importance it is considered to hold, and the lower each weight, the less importance. Weight 1, W_1, is applied to detection coverage, Weight 2, W_2, is applied to activity cost, and Weight 3, W_3, is applied to false positives. If the organization is more concerned with having a higher detection coverage and realizes that in order to have this it must tolerate a higher activity cost and a higher false positive count, then it can give a higher weighted value to W_1 and a lower weighted value to W_2 and W_3. For example an organization in which auditing is so critical that missing a detection could cause death or serious economic loss, such as a nuclear facility or a financial institution, would probably put more weight on W_1 and less weight on W_2 and W_3. If the organization is more concerned with having a lower total cost and

number of false alarms, perhaps because it has limited resources for robust auditing, and realizes that it will get less detections as a result, then it will put a lower weight on W_1 and a higher weight on W_2 and W_3.

For example, suppose a given organization created a set of 15 malicious scenarios contained in $SC_{M,}$ in a current audit switch, S_x, and they also executed a set of 15 minutes of non-malicious scenarios contained in $SC_{N,}$ and calculated their *CNUC*. After doing their calculations using the given metrics, they had a *CDC* of 0.667, a *CAC* of 3220, a *TAC* of 6906, and a *CFPC* of 34. If they decide that detections are very important to them, and costs and false positives are less important and they give W_1 a weight of 10, W_2 a weight of 2, and W_3 a weight of 2. The Switch Figure of Merit, *SFOM*, for that particular switch would be:

$$SFOM\big|_{S_x} = 10 \text{ x } (.667) - 2\left[\frac{3220}{6906}\right] - 2\left[\frac{34}{587}\right] = 5.6216 \qquad \text{(Eq. 3.16)}$$

The organization would calculate this *SFOM* for all switches, and then use the results to assist in selecting the switches for their auditing template based on the switches with the highest *SFOM*'s.

If the organization decided that detections are less important, and lower costs and false positives is more important and they give W_1 a weight of 2, W_2 a weight of 10, and W_3 a weight of 10. The Switch Figure of Merit, *SFOM*, would be:

$$SFOM\big|_{S_x} = 2 \text{ x } (.667) - 10\left[\frac{3220}{6906}\right] - 10\left[\frac{34}{587}\right] = -3.9078 \qquad \text{(Eq. 3.17)}$$

Since in this example the higher weights were given to W_2 and W_3, which are subtracted from W_1, it is possible for the *CFOM* to be a negative number. In this case the

organization would compare all switches *SFOM*'s, and the switches with the least negative number (higher of the two) would be more beneficial to the organization.

It is important to point out that when the organization chooses their weights for W_1, W_2, and W_3, they will need to be applied to all switches equally. If the organization chooses to change those weights, they must change them for all switches.

3.11.2.8. Switch Selection Criteria

Now that all results have been evaluated and calculations made, the organization can select the switches for its auditing template. An organization can select criteria to use based on its security requirements to select audit switches for its auditing template using the calculated *SFOM*'s as needed for decision making. These are the organizations Switch Selection Criteria (*SSC*). During this experiment, based on the simulated organizational security assumptions made earlier, the switches for the auditing template were selected based on the following criteria:

1. The switches are selected so that all scenarios are detected, regardless of the *SFOM* value.
2. If more than one switch detects the same scenarios, select the switch for auditing with the highest *SFOM* value, and do not select the others.
3. If a switch did not detect any scenario, then do not select it for auditing.

These criteria were used in conjunction with the results and calculations to create an auditing template. The results and the auditing template selected for this experiment are discussed in chapter 4.

3.11.3. Configuration Evaluation Method

The Audit Selection Criteria mentioned above, using the *SFOM* appears to be sufficient at identifying the best switches to enable for an auditing template, but can be considered naïve because it evaluates switches individually based upon their own merit, but doesn't compare sets of them together [3]. Another way of identifying the best configuration to identify a given set of malicious activities is to evaluate each possible audit configuration in its entirety. This is done using the Configuration Evaluation Method (*CEM*). This method examines all possible audit configurations available, which for the 18 switches is 2^{18}, or 262,144 possible configurations. It would be impossible to do this manually, but it can be done exhaustively using software. A software program developed using C++ for this experiment exhaustively takes each individual audit configuration available, calculates a Configuration Figure of Merit (*CFOM*) metric, and presents the best auditing configuration for detecting the given scenarios based on the switches with the highest *CFOM* value. Additionally, it calculates and indicates the *CFOM* for 3 available auditing configurations in existing security templates; *NIST EC*, *NIST SSLF*, and *NSA SNAC EC*. Sample output from this program can be seen in Appendix E, and it's written code can be seen in Appendix F.

3.11.3.1. Configuration Figure of Merit

Just like the *SFOM*, the Configuration Figure of Merit (*CFOM*) is a weighted metric used for objective comparison. It is used to compare multiple audit configurations. It applies a value to each configuration based on its performance given the selected

scenarios. It allows the configurations to be objectively compared against each other. To calculate the *CFOM*, the calculations from the following metrics are used:

1. Configuration Detection Coverage (*CDC*)
2. Configuration Activity Cost (C*AC*)
3. Configuration Total Activity Cost (C*TAC*)
4. Configuration False Positive Count (C*FPC*)

Just as with the Switch Evaluation Method (*SEM*), in order to represent each of the metrics and the switch figure of merit, the problem is loosely framed in the context of set theory [38]:

Auditing configuration is determined by setting or clearing a set of 18 auditing switches. Each auditing switch takes on one of two discrete values, either on or off:

$$Value = \{0,1\} \quad \text{(Eq. 3.18)}$$

A Switch, S_x where $\underline{x} = [1..M]$, is assigned an element of the set *Value*:

$$S_x = Value \quad \text{(Eq. 3.19)}$$

An auditing configuration, *AC*, consists of a set of *M* Switches. When an auditing configuration is evaluated in it's entirety based on its current set of *M* switches, it is referred to as AC_X:

$$AC_x = \{S_1, S_2, ..., S_M\} \quad \text{(Eq. 3.20)}$$

The scenarios, *SC*, consist of a malicious scenario set, SC_M, a non-malicious scenario set, SC_N. The malicious scenario set, SC_M, consists of the 18 selected simulated malicious scenarios:

$$SC_M = \{SC_{M_1}, SC_{M_2}, ..., SC_{M_{18}}\} \quad \text{(Eq. 3.21)}$$

The non-malicious scenario set, SC_N, consists of one 15 minute scenario consisting of simulated typical normal user activity:

$$SC_N = \{SC_{N_1}\} \quad \text{(Eq. 3.22)}$$

The events are the events that appear in the security audit log for a after the scenarios are executed. They consist of the events from the malicious scenarios, E_M, and the events from the non-malicious scenarios, E_N. The total events for a given switch, S_x, after all malicious scenarios, E_M, and non-malicious scenarios, E_N, have been executed are E_X where $(x \mid x \in \{S_1, S_2, ..., S_M\})$. The set of all possible events is the total of all events for all switches, E_T, as $E_T = \bigcup E_x$ where $(x \mid x \in \{S_1, S_2, ..., S_M\})$.

3.11.3.2. Configuration Detection Coverage

The Configuration Detection Coverage (*CDC*) is a measure of how well a complete auditing configuration performs at detecting a set of malicious scenarios, SC_M. It is calculated the same way that the *SDC* is calculated, with the exception that the results are based on a given entire audit configuration, AC_x, instead of a single audit switch, S_X. The set of malicious scenarios detected, SC_D, where $SC_D \subseteq SC_M$. It is calculated as the sum of detected scenarios, SC_D, divided by the total number of malicious scenarios, SC_M, in the malicious scenario set for a given audit configuration, $AC_x = \bigcup AC_T \ (T \mid T \in \{S_1, S_2, ..., S_M\})$.

$$CDC\big|_{AC_x} = \frac{|SC_D|}{|SC_M|} \quad \text{(Eq. 3.23)}$$

The C*DC* is an unbiased indicator as to how well a particular Audit Configuration, AC_x, is at detecting the malicious behaviors contained in a given malicious scenario set, SC_M. For example, suppose an organization has created a set of 15 malicious scenarios in order to determine the effectiveness of its audit configuration. They would run each of the scenarios contained in the malicious scenario set, SC_M, and determine which of the scenarios in the set are detected. If 13 of the 15 total scenarios are detected, the calculation would be:

$$CDC\big|_{AC_x} = \frac{|SC_D|}{|SC_M|} = \frac{13}{15} = 0.867 \qquad \text{(Eq. 3.24)}$$

While *CDC* is an effective measure of the detection coverage for a given audit configuration under a given set of scenarios, it fails to account for the cost in log file size and false positives encountered.

3.11.3.3. Configuration Activity Cost

As noted earlier, each event, or line in a log file, incurs a cost in terms of disk space and needs to be considered for the entire audit configuration. Configuration Activity Cost (*CAC*) is the total activity cost for the current audit configuration, which is the sum of all events in the security log for that configuration after the malicious scenarios have been executed and the switch *CNUC* is calculated. It is calculated as the sum of the *SAC's* for each current audit switch enabled for auditing: $AC_x = \bigcup AC_s$ $\left(s \mid s \in \{S_1, S_2, ..., S_M\}\right)$.

$$CAC\big|_{AC_x} = |E_M| + CNUC|_{AC_x} \qquad \text{(Eq. 3.25)}$$

For example, suppose an organization has created and executed its set of 15 malicious scenarios contained in SC_M, and calculated enabled all enabled switches *CNUC's* in order to determine the effectiveness of its audit configuration. They currently have 7 audit switches enabled for auditing (S_2, S_3, S_4, S_5, S_7, F_7, and S_8) and their *SAC*s are 322, 193, 278, 256, 3023, 165, and 54 respectively. The *CAC* calculation would be:

$$CAC\big|_{AC_x} = \begin{pmatrix} 322+193+278+256 \\ +3023+165+54 \end{pmatrix} = 4291 \qquad \text{(Eq. 3.26)}$$

3.11.3.4. Configuration Total Activity Cost

The Configuration Total Activity Cost (*CTAC*), is identical to the Total Activity Cost, *TAC*, used in the *SEM*. It is the total number of activity costs for all switches, S_T, not just the ones enabled in the given audit configuration, AC_X. It is the total of all switches *SAC*'s. It is the set of all possible events, or the total of all events for all switches, E_T, as $E_T = \bigcup \; E_x$ where $\left(x \mid x = \{S_1, S_2, ..., S_M\}\right)$. It is calculated as the sum of the *SAC*'s for all switches, or the union of all switch total events, E_T:

$$CTAC\big|_{AC_x} = E_T = \bigcup \; E_x \qquad \text{(Eq. 3.27)}$$

For example, suppose an organization enabled all switches for auditing, created and executed a set of 15 malicious scenarios contained in SC_M, and calculated each switches *CNUC* for SC_N. All *SAC*'s are as follows:

Switch	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9
SAC	1320	543	36	1003	920	45	890	63	0	943	54	5	0	0	31	133	920	0

The calculation would be:

$$CTAC\big|_{AC_x} = \begin{pmatrix} 1320+543+36+1003+920 \\ +45+890+63+0+943+54 \\ +5+0+0+31+133+920+0 \end{pmatrix} = 6906 \quad \text{(Eq. 3.28)}$$

3.11.3.5. Configuration False Positive Count

Just as with the *SEM*, the S*FPC* is a measure of how many critical events occurred that did not correspond to a malicious scenario, only for the *CEM* it's determined for the given audit configuration, AC_X. It is calculated as the sum of critical events not detecting a malicious scenario that occurred in the log file resulting from the set of malicious scenarios, and the *CNUC*. It is simply a count of events that have the same event numbers as detection but are not detections in the *SAC*. This is not easily defineable by set theory:

For example, suppose an organization evaluating their current audit configuration, AC_X, created a set of 15 malicious scenarios contained in $SC_{M,}$ calculated all *CNUC*'s for $SC_{N,}$ have 6 audit switches enabled for auditing (S_2, F_2, S_4, S_5, S_7, and S_8), and their *SFPC*s are 2, 33, 212, 0, 510, and 0 respectively. The *CFPC* calculation would be:

$$CFPC\big|_{AC_x} = (2+33+212+0+510+0) = 757 \quad \text{(Eq. 3.29)}$$

3.11.3.6. Configuration Figure of Merit

In order to evaluate an overall audit configuration's ability to detect the given scenarios so that the optimal auditing template can be determined, the above metrics are combined into a single, derived metric:

$$CFOM\big|_{AC_x} = W_1 \text{ x } CDC \; - \; W_2 \text{ x } \left[\frac{CAC}{CTAC}\right]\Big|_{AC_x} \; - \; W_3 \text{ x } \left[\frac{CFPC}{CAC}\right]\Big|_{AC_x} \quad \text{(Eq. 3.30)}$$

Just as with the *SFOM* discussed in section 3.11.2.7, the organization can place values into the weights W_1, W_2, and W_3 based on its current security requirements. For example, suppose an organization evaluating their current audit configuration, AC_X, created a set of 15 malicious scenarios contained in SC_M, calculated all *SNUC*'s for non-malicious scenarios in SC_N, and have 8 audit switches enabled for auditing (S_2, F_2, S_4, S_5, F_5, S_6, S_7, and S_8). After completing their calculations using the given metrics, they have a *CDC* of .867, a *CAC* of 4291, a *CTAC* of 6906, and a *CFPC* of 757. If they decide that detections are very important to them, and costs and false positives are less important and they give W_1 a weight of 3, W_2 a weight of 1, and W_3 a weight of 1. The Configuration Figure of Merit, *CFOM*, would be:

$$CFOM\big|_{AC_x} = 3 \text{ x } (.867) - 1\left[\frac{4291}{6906}\right] - 1\left[\frac{757}{4291}\right] = 1.803 \qquad \text{(Eq. 3.31)}$$

If they decide that detections are less important, and lower costs and false positives is more important and they give W_1 a weight of 1, W_2 a weight of 3, and W_3 a weight of 2. The Configuration Figure of Merit, *CFOM*, would be:

$$CFOM\big|_{AC_x} = 1 \text{ x } (.867) - 3\left[\frac{4291}{6906}\right] - 2\left[\frac{757}{4291}\right] = -1.350 \qquad \text{(Eq. 3.32)}$$

These two values of 1.803 and -1.350 have no relationship to each other, because they are based on different weights. They would be used to compare against other *CFOM*'s which have been calculated using their same weights. Comparisons can only be made between configurations if the same weights are used for each configuration.

Based on the weights for W_1, W_2, and W_3 that an organization decides upon, the software calculates the *CFOM* for all possible audit configurations, and then uses the

results to select the 20 audit configurations with the highest *CFOM*'s. It also displays the *CFOM* that would have resulted from the use of the audit configurations in 1 of 3 security templates; *NIST EC*, *NIST SSLF*, and *NSA SNAC*.

3.11.3.7. Existing Configuration Evaluation

The calculations used for the *CEM* can be used to evaluate a current auditing template against an existing audit configuration in a security template (e.g. *NIST EC*, *SNAC*, etc.), or to compare between two audit configurations. In this experiment, Microsoft Excel® is used to input audit configuration data and calculate the resulting *CFOM*. An example of this is performed in Chapter 4, and the spreadsheet can be seen in Appendix 3.2.

3.12. Summary

The experimental design described in this chapter can be used to create a customized insider threat auditing template for a Microsoft Windows XP®. It is used to determine how well a Microsoft Windows XP® auditing system can detect particular insider threats, using scenarios focused on unique organizational security requirements. The approach, auditing system, and evaluation technique were covered. The results of the experiment are presented in the next chapter.

IV. Results

4.1. Chapter Overview

This chapter presents the results of the experimental methodology applied in Chapter 3. First, the results of the Switch Evaluation Method (*SEM*) are analyzed using the Switch Figure of Merit (*SFOM*) to determine which switches achieve the best results based on selected scenarios. Second, the results and the Switch Selection Criteria (*SSC*) are used to select an auditing template. Using the Configuration Evaluation Method (*CEM*) the auditing template is subsequently compared to audit configurations from the following security templates: (1) National Institute of Standards and Technology's (*NIST*) Enterprise Client Security Template (*NIST EC*), (2) N*IST* 's Specialized Security Limited Functionality Security Template (*NIST SSLF*), and (3) National Security Agency Systems and Network Attack Center's Security Template (*NSA SNAC*) [30, 31]. Finally, using the developed software, the *CEM*'s ability to create an auditing template is demonstrated based on the same results of the malicious and non-malicious scenarios used in the *SEM*.

4.2. Switch Evaluation Method Results

The Switch Evaluation Method (*SEM*) results are presented in this section. The *SEM* gives the information owner the ability to create an auditing template using the individual Switch Figures of Merit (*SFOM*), and the Switch Selection Criteria (*SSC*). All results used to calculate the *SFOM* are presented, as well as the final *SFOM* results and the final resulting auditing template. This auditing template is then compared with the

audit configuration of the following security templates: (1) *NIST EC*, (2) *NIST SSLF*, and (3) *NSA SNAC*.

4.2.1. Detections

Detections consist of the events that are present in the security log that identify a malicious action after the set of malicious scenarios is simulated. As seen in Table 4.1 and Figure 4.1, of the 18 malicious scenarios 17 were detected. The detections are in bold to indicate which switch detected each particular scenario. When the malicious scenarios are listed, the "a" represents the misuse of administrator privileges, and the "u" represents normal user privileges. The detailed malicious scenario results can be found in Appendix D.

Based on the results, here are some notable findings:

1. Scenario 11u is the only scenario not detected.
2. Switch S_5 has the greatest number of detected scenarios with 7
3. Switch S_2 has the second most detections with 6
4. Switches S_1, S_4, F_5, S_8, and S_9 has the third most detections with 3 each.
5. There were several scenarios detected by single switches:
    a. Scenarios 5a, 6a, 7a, and 8a are only detected by switch S_2.
    b. Scenarios 2a, 3a, 9a, 12u, 14a, and 15a are only detected by switch S_5.
    c. Scenarios 4a, 10a, and 13u are only detected by switch F_5.
    d. Scenario 17u is only detected by switch S_9.
    e. Scenario 18u is only detected by switch S_8.
6. There were several scenarios detected by multiple switches:
    a. Scenarios 1a, 15a, and 16u are all detected by both switches, S_1 and S_4.
    b. Scenarios 1a and 15a are both detected by switches S_1, S_2, S_4, S_5, S_6, S_7, S_8, and S_9.
7. There are several switches that didn't detect any of the scenarios:

a. Switches F_1, F_2, S_3, F_3, F_4, F_7, F_8, and F_9 detected no scenarios.

Table 4.1: Total Detections By Switch, and Scenario Detection Coverage

	Switches (Categories)																	
Scenarios	**S1**	**F1**	**S2**	**F2**	**S3**	**F3**	**S4**	**F4**	**S5**	**F5**	**S6**	**F6**	**S7**	**F7**	**S8**	**F8**	**S9**	**F9**	**Detected**
1a	**1**	0	**1**	0	0	0	**1**	0	**1**	0	**1**	0	**4**	0	**2**	0	**2**	0	**Yes**
2a	0	0	0	0	0	0	0	0	**15**	0	0	0	0	0	2	0	0	0	**Yes**
3a	0	0	0	0	0	0	0	0	**62**	0	0	0	0	0	2	0	0	0	**Yes**
4a	0	0	0	0	0	0	0	0	0	**108**	0	0	0	0	0	0	0	0	**Yes**
5a	1	7	**2**	0	0	0	6	6	86	0	0	0	1	0	35	0	0	0	**Yes**
6a	3	7	**6**	0	0	0	11	7	21	0	0	0	11	0	88	0	0	0	**Yes**
7a	2	8	**1**	0	0	0	7	8	33	0	0	0	10	0	47	0	0	0	**Yes**
8a	0	1	**1**	0	0	0	9	2	17	0	0	0	10	0	46	0	0	0	**Yes**
9a	3	2	0	0	0	0	2	1	**6**	0	0	0	5	0	12	0	0	0	**Yes**
10a	0	1	0	0	0	0	2	0	0	**1**	0	0	0	0	12	0	0	0	**Yes**
11u	1	0	0	0	0	0	2	1	0	0	0	0	1	0	10	0	0	0	**No**
12u	0	0	0	0	0	0	0	0	**54**	0	0	0	0	0	3	0	0	0	**Yes**
13u	0	0	0	0	0	0	2	0	0	**108**	0	0	0	0	9	0	0	0	**Yes**
14a	0	1	0	0	0	0	4	1	**76**	78	0	0	13	0	22	0	0	0	**Yes**
15a	**2**	1	**1**	0	0	0	**3**	1	**5**	0	**1**	0	**6**	0	**13**	0	**1**	0	**Yes**
16u	**1**	4	0	0	0	0	**9**	4	12	3	1	0	39	0	40	0	28	0	**Yes**
17u	2	0	0	0	0	0	35	0	73	0	12	0	22	0	36	7	**16**	0	**Yes**
18u	1	1	0	0	0	0	2	1	0	0	11	0	8	0	**30**	0	0	0	**Yes**
Total Detects	**3**	**0**	**6**	**0**	**0**	**0**	**3**	**0**	**7**	**3**	**1**	**0**	**1**	**0**	**3**	**0**	**3**	**0**	**17/18**

4.2.2. Analysis of Detections

As Table 4.1 indicates, Switch S_5 detected a large number of the scenarios. This is mainly because S_5 audits for successful object access, and most of the scenarios were based on the premise of attempting to access other users files. Switch S_2 detected a large number of scenarios also, mainly the ones in which administrator privileges were misused in some way. This switch audits successful account management in which accounts are created or changed, and is very useful to audit the actions of system administrators to ensure they are abiding by organization security policy. A few of the scenarios were detected by several switches, and most likely the switches with the best *SFOM* value will be selected, and the others not selected for the auditing template. Several switches did not log anything at all, and after being considered by the

organization, the decision will most likely be to not use them in the auditing template. Decisions to include switches based on these detection findings are based on the organizations Switch Selection Criteria (*SSC*).

Figure 4.1: Scenario Detections By Switch

4.2.3. Costs

Costs are obstructions to identifying the events that indicate a malicious act. They are events captured that are not indicative of a malicious act. They conceal and obfuscate the events necessary to identify a malicious act. They consist of Activity Costs (*AC*), and False Positive Events (*FPE*).

4.2.3.1. Malicious Scenario Activity Cost

The Malicious Scenario Activity Costs (*MSAC*) is the total number of events logged upon completion of simulating the malicious scenarios. As indicated by Table 4.2, there is a wide variety of scenario activity costs.

Here are some notable findings:

1. Switch S_5 has the highest Scenario Activity Cost at 461 events.
2. Switch S_8 has the second highest Scenario Activity Cost at 409 events.

3. Switch F_5 has the third highest Scenario Activity Cost at 298 events.
4. There are several switches with no Scenario Activity Cost:
    a. Switches F_2, S_3, F_3, F_7, and F_9 have no Scenario Activity Cost.

Table 4.2: Scenario Activity Costs and Calculated Normal User Costs

	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9
Malicious Scenario Activity Cost (*MSAC*)	17	33	12	0	0	0	95	32	461	298	26	0	130	0	409	7	47	0
NU Cost (15 minutes)	1	0	0	0	3	1	3	1	3	7	0	0	1	0	86	2	0	0
CNUC (40 hour week) = (NU Cost * 160)	160	0	0	0	480	160	480	160	480	1120	0	0	160	0	13760	320	0	0
SAC = (*MSAC* + *CNUC*)	177	33	12	0	480	160	575	192	941	1418	26	0	290	0	14169	327	47	0

4.2.3.2. Calculated Normal User Cost

The Calculated Normal User Cost (*CNUC*) is a representative of a weeks worth of non-malicious normal user activities. It is the total number of events logged after running the 15 minutes of non-malicious "normal user" behavior, multiplied by 160 to represent a week. As indicated by Table 4.2, once calculated there is noticeable range of Calculated Normal User Costs (*CNUC*). Here are some notable findings:

1. Switch S_8 has the highest *CNUC* at 13,760 events.
2. Switch F_5 has the second highest *CNUC* at 1,120 events.
3. Switches S_3, S_4, and S_5 have the third highest *CNUC* at 480 events.
4. There are several switches with no *CNUC*:
    b. Switches F_1, S_2, F_2, S_6, F_6, F_7, S_9, and F_9 have no *CNUC*.

4.2.3.3. Switch Activity Cost

The Switch Activity Cost (*SAC*) is the total number of events that appear in the security log after running the malicious scenarios (*MSAC)* as well as all calculated non-malicious normal user events (CNUC). It is calculated as the sum of the *MSAC* and the *CNUC*. Table 4.2 and Figure 4.2 indicates that the Switch Activity Costs (*SAC*) were extremely high in one switch, and significantly high in a couple others.

Here are the notable findings:

1. Switch S_8 has the highest *SAC* at 14,169 events.
2. Switch S_8 has the second highest *SAC* at 1,418 events.
3. Switch F_5 has the third highest *SAC* at 941 events.
4. Switches F_2, F_6, F_7 and F_9 have no *SAC*.

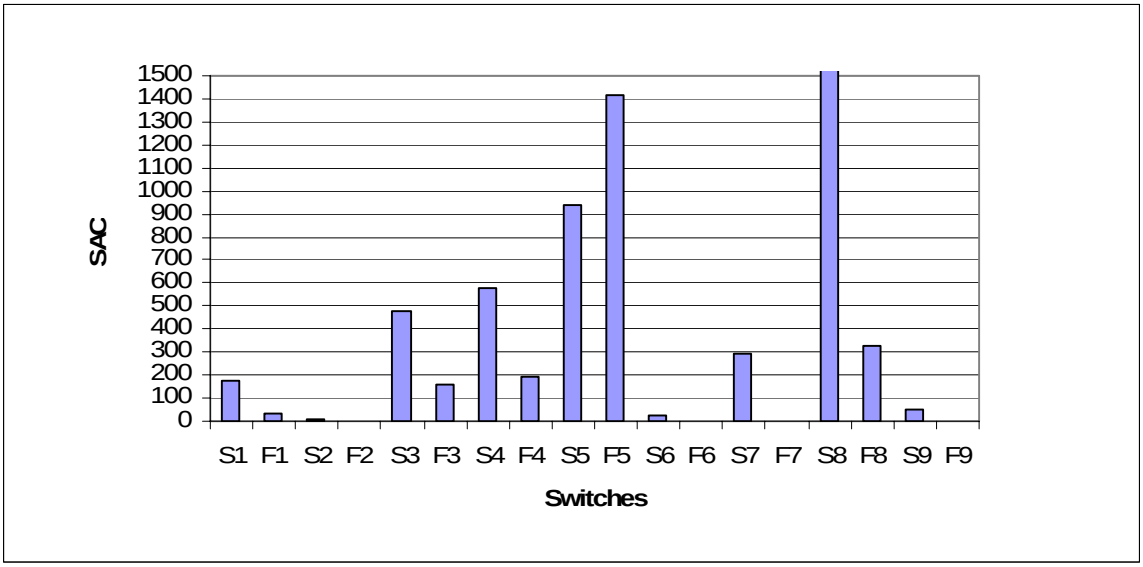

Figure 4.2: Switch Activity Cost (*SAC*)

4.2.3.4. Switch False Positive Count

False positives are events that appear to detect a malicious scenario because they have the same event number, but after evaluating them further it is determined that they do not. The Switch False Positive Count (*SFPC*) is the total number of false positives in the malicious scenarios and the *CNUC*. It is calculated as the sum of the malicious scenarios false positives and the *CNUC* false positives. As indicated by Table 4.3 and Figure 4.3, the range of Switch False Positive Counts (*SFPC*) closely resembles the range of *SAC*'s.

Here are the notable findings:

1. Switch S_8 has the highest *SFPC* at 6,296 events.

2. Switch F_5 has the second highest *SFPC* at 1,130 events.
3. Switch S_4 has the third highest *SFPC* at 350 events.
4. Switches S_1, F_1, S_3, F_3, S_6, F_6, F_7, F_8, S_9, and F_9 have no *SFPC*.

Table 4.3: Switch False Positive Count

	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9
Scenario False Alarms	0	0	2	0	0	0	30	28	27	10	0	0	7	0	216	0	0	0
NU False Alarms(15 mins)	0	0	0	0	0	0	2	1	1	7	0	0	1	0	38	0	0	0
NU False Alarms(40 hr wk) = (NU FA's * 160)	0	0	0	0	0	0	320	160	160	1120	0	0	160	0	6080	0	0	0
CFPC = (Scen FA + NU FA (40 hr wk))	0	0	2	0	0	0	350	188	187	1130	0	0	167	0	6296	0	0	0

4.2.3.5. Analysis of Costs and False Positives

As indicated by Table 4.2 and Figure 4.2, several of the switches had noticeably higher Switch Activity Costs (*SAC*) than the other switches, such as switches S_8 and F_5. Based upon this, depending on the number of detections that switch had (which will become evident in the *SFOM*), the organization may decide that it is not worth the cost to enable these switches. Consider switch S_8 for example. It is the only switch that detected scenario 18u, which is a user running a rootkit, but at an extremely large cost. Switch S_8 audits for successful startup and shutdown of processes. This enabled it to detect the process startup of the rootkit, but in order for this switch to be successfully implemented by an organization, a list of all authorized processes would need to be compared with the processes logged on a workstation, which is unfeasible. The organization's *SSC*, as well as the switches *SFOM* would assist in their decision to include it or not in their auditing template.

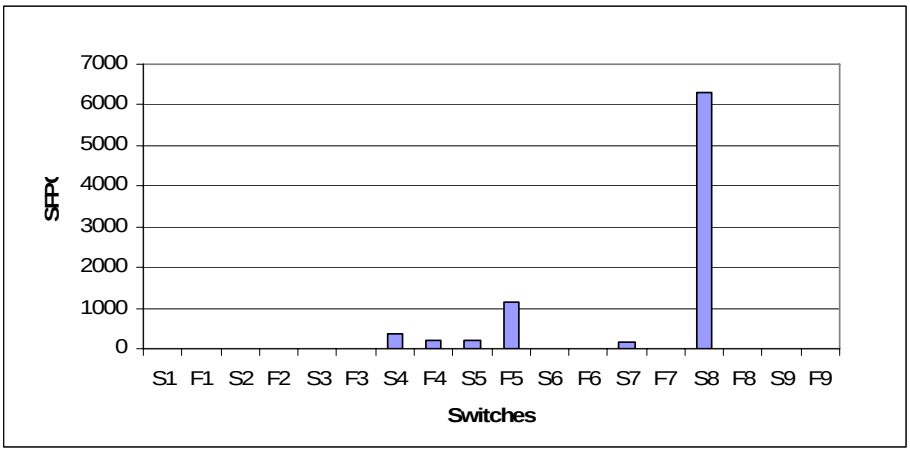

Figure 4.3: Switch False Positive Counts (*SFPC*'s)

The Switch False Positive Count (*SFPC*) was noticeably larger for several of the same switches that had the larger *SAC*'s, like S_8 and F_5. This makes these switches even less appealing to the organization, and would perhaps not be selected for their auditing template based on their *SSC*.

4.2.3.6. Switch Figure of Merit

The Switch Figure of Merit (*SFOM*) is calculated by using the formula presented in Chapter 3, as seen below:

$$SFOM\big|_{AuditSwitch_{CURRENT}} = W_1 * SDC - W_2 * \left[\frac{SAC}{TAC}\right] - W_3 * \left[\frac{SFPC}{SAC}\right]$$

An organization assigns weights to W_1, W_2, and W_3, based upon their security requirements and calculates a per switch *SFOM*. This *SFOM* in addition to the *SSC* help the information owner design an auditing template. When comparing the *SFOM*'s to one another, the higher the *SFOM* the better. Depending on the weights used for W_1, W_2, and W_3, the *SFOM* values can be negative. In comparing two negative *SFOM*'s, the least

negative value is greater, and will be the one to select between the two. In the last chapter, all Switch Figures of Merit (*SFOM*) were calculated using a Microsoft Excel Spreadsheet. A sample weight of W_1=5, W_2=1, W_3=1 is used to simulate a simulated security concern for a notional organization. After assigning the weights, the resulting *SFOM*'s are calculated and provided in Table 4.4.

Table 4.4: Switch Figure of Merit (*SFOM*), and Values used to Calculate Them

	SDC	SAC/TAC	SFPC/SAC	SFOM
S1	0.166666667	0.0093914	0	0.823941918
F1	0	0.0017509	0	-0.001750942
S2	0.333333333	0.0006367	0.16666667	1.499363294
F2	0	0	0	0
S3	0	0.0254682	0	-0.025468244
F3	0	0.0084894	0	-0.008489415
S4	0.166666667	0.0305088	0.60869565	0.194128847
F4	0	0.0101873	0.97916667	-0.989353964
S5	0.388888889	0.0499284	0.19872476	1.695791313
F5	0.166666667	0.0752374	0.79689704	-0.038801143
S6	0.111111111	0.0013795	0	0.554176026
F6	0	0	0	0
S7	0.111111111	0.0153871	0.57586207	-0.035693578
F7	0	0	0	0
S8	0.166666667	0.7517907	0.44435034	-0.362807745
F8	0	0.0173502	0	-0.017350241
S9	0.166666667	0.0024938	0	0.830839568
F9	0	0	0	0

4.2.3.7. Analysis of Switch Figures of Merit

As indicated by Table 4.4, switch S_5 has the highest *SFOM*, followed by S_2 and S_9. These *SFOM*'s will be used if needed for decision making by the information owner for creating an auditing template. Switches with zero values are the results of no detections for that switch.

4.2.3.8. Determining the Best Auditing Template

The best auditing template is created using the Switch Selection Criteria (*SSC*) and referring to Table 4.1 as well as the *SFOM* values in Table 4.4. The resulting auditing template is presented in Table 4.5. In accordance with the *SSC*, the switches are selected as follows:

1. Each switch that is the only switch to detect any given scenario, is enabled for auditing. This was the case with switches S_2, S_5, F_5, S_8, and S_9.
2. Each set of switches that detect the same scenarios are analyzed by their *SFOM* value, and the one with the highest value is selected for auditing and the others are not. S_1 and S_4 both detected scenarios 1a, 15a, and 16u. S_1 has a *SFOM* value of 0.8239 and S_4 has a *SFOM* value of 0.1941. S_4 has a higher *SFOM* so it is selected for audit, and S_4 is not. S_6 and S_7 also detected scenarios 1a and 15a, and since S_1, S_5, and S_8 also detect those scenarios and they were selected using criteria number 1, so S_6 and S_7 are not selected.
3. All remaining switches that do not detect any scenarios are not selected for auditing.

Table 4.5: Selected Audit Template

	SFOM	Selected Audit Template	Criteria Followed
S1	**0.82394**	**Audit**	Followed Criteria 2 and was selected
F1	**-0.00175**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S2	**1.49936**	**Audit**	Followed Criteria 1, Automatically selected
F2	**0.00000**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S3	**-0.02547**	**No Audit**	Followed Criteria 3 and was **NOT** selected
F3	**-0.00849**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S4	**0.19413**	**No Audit**	Followed Criteria 2 and was **NOT** selected
F4	**-0.98935**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S5	**1.69579**	**Audit**	Followed Criteria 1, Automatically selected
F5	**-0.03880**	**Audit**	Followed Criteria 1, Automatically selected
S6	**0.55418**	**No Audit**	Followed Criteria 2 and was **NOT** selected
F6	**0.00000**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S7	**-0.03569**	**No Audit**	Followed Criteria 2 and was **NOT** selected
F7	**0.00000**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S8	**-0.36281**	**Audit**	Followed Criteria 1, Automatically selected
F8	**-0.01735**	**No Audit**	Followed Criteria 3 and was **NOT** selected
S9	**0.83084**	**Audit**	Followed Criteria 1, Automatically selected
F9	**0.00000**	**No Audit**	Followed Criteria 3 and was **NOT** selected

4.2.3.9. Configuration Comparison

Now that an auditing template has been created using the *SEM*, it is evaluated using the Configuration Figure of Merit (*CFOM*) and compared with existing audit configurations. Using Microsoft Excel®, all *CFOM* calculations are performed and the newly created auditing template is compared with 3 preexisting audit configurations in security templates; *NIST EC*, *NIST SSLF*, and *NSA SNAC*. Based on the same sample weight used for the switch evaluation of W_1=5, W_2=1, and W_3=1, each configuration is calculated. The resulting *CFOM* values are presented in Table 4.6 and Figure 4.4. It is evident that based on the scenarios and the *SSC* that the custom audit configuration performs better than existing configurations.

Table 4.6: Results of Audit Configuration Comparison

Audit Configuration	CFOM Value
Created Auditing Template	3.378496594
NIST EC	1.757262389
NIST SSLF	2.25058253
NSA SNAC	2.25058253

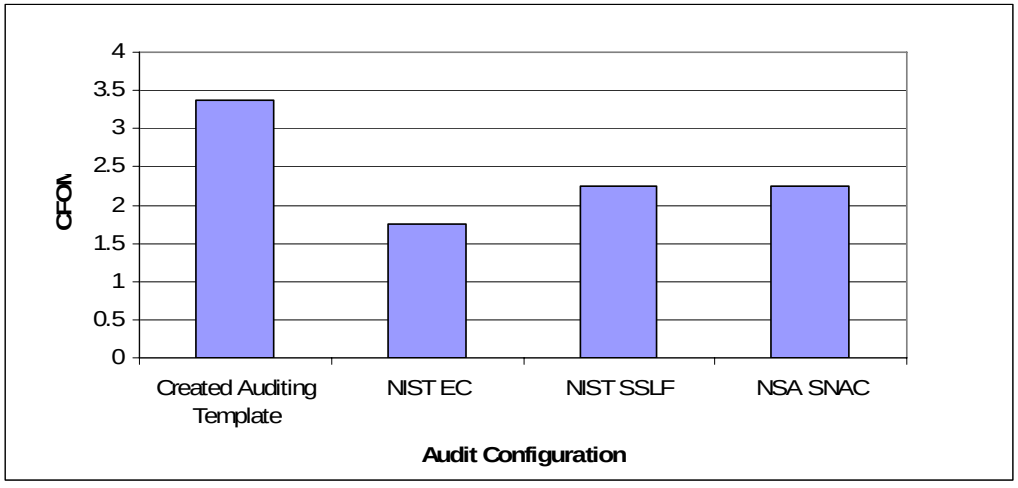

Figure 4.4: Comparison of Configuration Figures of Merit

4.3. Configuration Evaluation Method Results

The Configuration Evaluation Method (*CEM*) is used to create the best audit configuration based on an exhaustive comparison of all possible audit configurations *CFOM* values, using a program written in C++ for this experiment . The code can be seen in Appendix F. All results from the scenarios, as well as costs are input, just as with the Switch Evaluation Method, and the program uses the Configuration Figure of Merit (*CFOM*) calculations to select a list of best audit configurations. Many configurations were generated for each weight selected and are too numerous to list. They are provided in Appendix E. As seen in Table 4.7, when the weight of W_1=1, W_2=0, and W_3=0, the Configuration Detection Coverage (*CDC*) is the only value of importance in the formula, so switches were chosen based solely on the *CDC* value of 0.994 (17 out of 18 scenarios detected). After experimenting with the weights, it is noticed that by giving the weights a value of W_1=1, W_2=0.07, and W_3=0, would give enough weight to W_2 for a configuration to be selected with a reduced Configuration Activity Cost (*CAC*) value of 16,764 while still maintaining the same *CDC* value. This is an example of how to adjust the weights to achieve a lower cost while still maintaining the same detection level. When the weights were changed to W_1=1, W_2=0.08, and W_3=0.08, the weights associated with *CAC* and *CFPC* were raised enough for the custom audit configurations to have a significantly lower *CAC* and *CFPC* values at the expense of the *CDC* which was lowered to 0.889 (16 out of 18 detections). This shows that when the weights of W_2 and W_3 were raised enough, one detection was not considered because the cost associated with it was too high, but the costs went down significantly. Achieving an

acceptable threshold is something that an organization would consider if they wanted to significantly reduce costs and false positives at the expense of detections.

Table 4.7: Weight Affect on Results for Configuration Evaluation

W1	W2	W3	CDC	CAC	CFPC
1	0	0	0.944	17997	8320
1	0.07	0	0.944	16764	7615
1	0.08	0.08	0.889	3595	1319

4.4. Summary

The two methods differ in their approaches. The Switch Evaluation Method (*SEM*), is based on decision making from the Switch Selection Criteria (*SSC*) and uses the Switch Figure of Merit (*SFOM*) to assist in making those decisions. It bases the *SFOM* on individual switch statistics. The Configuration Evaluation Method (CEM), uses the Configuration Figure of Merit (*CFOM*) to evaluate each audit configuration possible. It is based exclusively on calculations, and needs to be fine tuned to understand the impact of adjusting the weights for W_1, W_2, and W_3 based on the selected scenarios. Once the weights have been understood, the *CEM* results in a more thorough evaluation of the audit configuration and produces more conclusive results, mainly because it does an exhaustive evaluation of each possible configuration. The results from both change dramatically based upon the malicious scenarios that are selected for the test. Once each method is understood and configured within an organization, they can produce similar results. Based on the scenarios selected in this experiment, once the weights were fine-tuned, the *CEM* appeared to give a more thorough and complete end result. It is recommended that an organization try both methods to determine the one that suits them best based upon their specific security guidelines.

V. Conclusions

5.1. Chapter Overview

This chapter summarizes the research efforts and the research goals. First, the impact of this research and its utility are discussed. Next, ways that it can be implemented and follow on suggestions are proposed. Finally, the chapter ends with a discussion of potential future research efforts.

5.2. Restatement of Research Goal

The research goal was to develop a methodology for creating a customized insider threat auditing template based on unique organizational security requirements. This was performed by creating metrics that were used in two separate methods for creating this auditing template.

5.3. Research Impact

Application of the methodology developed in this research provides an organization with a method to optimize its auditing capabilities. This increases the organizations capability to detect and identify malicious actions early enough to mitigate any potential damages. It can also reduce the workload required to review the audit logs by reducing their size. Additionally, it can increase the efficiency of security administrators ability to detect malicious activity when reviewing audit logs by maximizing the amount of useful information they contain. Finally, it allows an organization to be aware of what is being audited within its enterprise network and the effectiveness of the ability to detect malicious acts.

5.4. Suggested Implementaion

Rather than rely upon one-size-fits-all auditing configurations available in recommended security templates, it is suggested that an organization identify its specific insider threat security requirements and use the methodology presented in this research to create its own custom auditing template. It is suggested that both methods proposed be evaluated by the organization to determine the best method for satisfying those identified security requirements. It would be very beneficial for the organization to know what is being audited within its organizational workstations and to ensure that the auditing that is being performed complies with its security requirements. Because security requirements change, it is suggested that once the organization has initiated this methodology into its security policy, that it periodically use it to reevaluate its auditing template to ensure that it still fits their security needs. What is presented in this research is an example. It is suggested that an organization use this example as a baseline and make it as robust as possible by creating numerous simulated malicious scenarios and using more accurate normal user statistics. The more robust the organization makes this methodology, the better the results. The C++ program created for this research is provided in Appendix F, and can be modified to give the tailored output that the organization needs to make its decisions on an auditing template.

5.5. Future Research

There are three areas in which this research can be expanded. The first area of future research is to apply the methodology to networked workstations and servers to identify and customize auditing of network related communication within an organization.

A second area is to expand the methodology to other networked systems that perform various other roles such as a servers, routers, and switches. By applying the methodology in various other capacities, it can achieve an improved awareness of the auditing situations in those areas and allows for the optimization of auditing capabilities.

A third area is in applying the methodology to outside threats. Currently many organizations apply auditing configurations available in widely used security templates without knowing what they audit for and if they are effective. If this research is applied to outside threats then perhaps their auditing can be optimized as well.

5.6. Summary

This research provides information owners a methodology for maximizing their auditing capabilities within their organization. It provides the groundwork for building an auditing template and for evaluating an existing auditing configuration to determine if it meets the needs of the organizations identified security requirements. The potential exists for the functionality to be used in a wide array of organizations.

Appendix A: Simulated Malicious Insider Scenarios

Table A.1: Simulated Malicious Insider Scenarios

1a	Attempt to Circumvent Auditing (clear Audit Logs) as Admin
2a	Attempt to access users folder as Admin (Users Folder has no *SACL*, but file inside does. Admin has access to folder and file)
3a	Attempt to access a users file successfully as admin with *SACL* (Admin has access to file, auditing is enabled in *SACL*)
4a	Attempt to access a users file unsuccessfully as admin with *SACL* (Admin Does NOT have access to file, auditing is enabled in *SACL*)
5a	Attempt to access a users file as admin with *SACL* by changing users password and logging in as the user
6a	Attempt to access a users file as admin by creating an admin account to use to mask attempts
7a	Attempt to access a users file as admin by changing a user account to admin to use to mask attempts
8a	Attempt to access a users file by changing a user account to admin by placing into admin group to use to mask attempts
9a	Attempt to access a users file as admin by attempting to create a hardlink
10a	Attempt to access a users file as admin with an *SACL* by attempting to create a hardlink
11u	Attempt to access another users folder as user (Users Folder has no *SACL*, but file inside does. Admin has access to folder and file)
12u	Attempt to access another users file as user successfully with *SACL* (User has access to file, Auditing is enabled in *SACL*)
13u	Attempt to access another users file as user unsuccessfully with *SACL* (User does not have access to file, Auditing is enabled in *SACL*)
14a	Attempt to access a users file as admin with *SACL* by changing ownership of the file
15a	Attempt to access a users file as admin by attempting to Change Security Policy
16u	Attempt to guess Admin password by rebooting into safemode and guessing admin password
17u	Access a users file by booting with a boot cd and deleting password
18u	Run a rootkit

Note: a = administrator privilege, u = normal user privilege

Appendix B: Simulated Malicious Insider Scenarios Script

Simulated Malicious Scenarios Script (With Confidentiality a Primary Concern, Attempting to Access Another Users files)

Open the folder to save all data for the scenario so that you remember what scenario you're doing
Log in as Levoy
Go to Start, Control Panel, Administrative Tools, Local Security Policy, Audit Policy, and Set Appropriate
Switch to Success and Failure, all others to "No Auditing"

1a. Attempt to Circumvent Auditing (clear Audit Logs) as Admin
Clear all audit logs

2a. Attempt to access users folder as Admin (Users Folder has no SACL, but file inside does. Admin has access to folder and file)
Browse to "C:\Documents and Settings\User 2\User 2's Documents\User 2.txt" folder, then close it

3a. Attempt to access a users file successfully as admin with SACL (Admin has access to file, auditing is enabled in ACL)
Browse to C:\User 2 (with Admin and User 4 Access)\User 2.txt, then close it

4a. Attempt to access a users file unsuccessfully as admin with SACL (Admin Does NOT have access to file, auditing is enabled in ACL)
Browse to C:\User 2 (Only User 2 Access), It should say "Access Denied", then click OK

5a. Attempt to access a users file as admin with SACL by changing users password and logging in as the user
Browse to Control Panel, User Accounts, and Change the Password for User 2 to "password"
Log off as "levoy" and log on as "User 2" with the new password you changed
Browse to C:\User 2 (Only User 2 Access)\User 2.txt, then close it
Log off as "User 2" and back in as "levoy"

6a. Attempt to access a users file as admin by creating an admin account to use to mask attempts
Browse to Control Panel, User Accounts, and Create an Account Called "Admin" with Admin privileges
Log off as "levoy" and log on as "Admin Account"
Browse to C:\Documents and Settings\User 2\My Documents\User 2.txt, then close it
Log off as "Admin Account" and log on as "levoy"
Delete "Admin" Account and all it's files

7a. Attempt to access a users file as admin by changing a user account to admin to use to mask attempts
Browse to Control Panel, User Accounts, and Change User 1 Account to have Admin Privileges
Log off as "levoy" and log on as "User 1"
Browse to C:\Documents and Settings\User 2\My Documents\User 2.txt, then close it
Log off as "User 1" and log on as "levoy"

8a. Attempt to access a users file by changing a user account to admin by placing into admin group to use to mask attempts
Browse to Control Panel, Administrative tools, Computer management, Local Users and Groups, Groups, Administrator, and Add User 3 to the Admin Group
Log off as "levoy" and log on as "User 3"
Browse to C:\Documents and Settings\User 2\My Documents\User 2.txt, then close it
Log off as "User 3" and log on as "levoy"

9a. Attempt to access a users file as admin by attempting to create a hardlink
Open a Dos prompt by going to All Programs, Accessories, Command Prompt
At the C:\ Prompt Type "fsutil hardlink create test "C:\Documents and Settings\User 2\My Documents\User 2.txt", it should say Hardlink created
Type edit test, it should say "test"

10a. Attempt to access a users file as admin with an SACL by attempting to create a hardlink
Open a Dos prompt by going to All Programs, Accessories, Command Prompt
At the C:\ Prompt Type "fsutil hardlink create test2 c:\User 2 (Only User 2 Access)\User 2.txt"
It should say, "Access Denied"

11u. Attempt to access another users folder as user (Users Folder has no SACL, but file inside does)
Log off as "levoy" and log on as "User 4"
Browse to "C:\Documents and Settings\User 2" folder, then close it

12u. Attempt to access another users file as user successfully with SACL (User has access to file, Auditing is enabled in ACL)
Browse to C:\User 2 (with Admin and User 4 Access)\User 2.txt, then close it

13u. Attempt to access another users file as user unsuccessfully with SACL (User does not have access to file, Auditing is enabled in ACL)
Browse to C:\User 2 (Only User 2 Access), then click OK

14a. Attempt to access a users file as admin with SACL by changing ownership of the file
Log off as "User 4" and log on as "levoy"
Browse to C:\User 2 (Only User 2 Access), right click on it and change ownership to levoy
Then Go to the same folder, right click, and add levoy to have full access to it.
Then Browse to C:\User 2 wo Admin Access\User 2.txt, then close it

15a. Attempt to access a users file as admin by attempting to Change Security Policy
Log off as "User 4" and log on as "levoy"
Go to Start, Control Panel, Administrative Tools, Local Security Policy, Audit Policy, and Set all to "No Auditing"
Attempt to browse to both C:\User 2 (with Admin and User 4 Access)\User 2.txt, and C:\User 2 wo Admin Access (for 15a)
Categories 1 and 3 are not used in workstation, Account logon events and Directory Services
Save all logs to desktop as .evt and .csv and then drag them to appropriate folder
Shut Down the VMWare Computer
Make Sure "Snapshot 1" is loaded

Additional Scenarios
Log on as levoy, then clear audit log

16u. Attempt to guess Admin password by rebooting into safemode and guessing admin password
Reboot virtual machine into safe mode
Make 3 unsuccessful attempts to log onto the administrator account with the password "pass"
Make one successful attempt to log onto the administrator account with the password "password"
Reboot back into normal mode

17u. Access a users file by booting with a boot cd and deleting password
Reboot with boot cd in
Change User 2 password to null
Reboot without boot cd in and log onto User 2 account

Browse to C:\User 2 (Only User 2 Access)
Log off as User 2

18u. Run a rootkit
Log on as levoy
Insert pen drive
Run furoot
Eject pendrive

Save all logs to desktop as .evt and .csv and then drag them to appropriate folder
Shut Down the VMWare Computer
Make Sure "Snapshot 1" is loaded

Appendix C: Simulated Non-malicious Insider (Normal User) Scenarios Script

Note: This Simulated Non-malicious Insider (Normal User) Script took 15 minutes to run:

Normal User Activity Script

1. Log off as "levoy" and log on as "User 4"
2. Open Word
3. Type the word "text"
4. Save the document to the desktop
5. Close Word
6. Open Excel
7. Type the word "text" into the first cell
8. Save the document to the desktop
9. Close Excel
10. Open Internet Explorer
11. Try to browse to a web page
12. Close Internet Explorer
13. Open Solitaire and play one hand then exit
14. Open Media player and play one of the sample songs then exit
15. Right Click and Create a new text document on the desktop
16. Right Click and Create a new wave sound on the desktop
17. Right Click and Create a new bitmap image on the desktop
18. Delete these 3 files
19. Right click on the desktop, go to properties, settings, then change the desktop picture and preview a screensaver
20. Insert a CD and browse to a file on it
21. Eject the CD
22. Insert a Pen Drive
23. Right Click on the desktop and create a new word document
24. Type "text"
25. Save the document to the desktop
26. Save the document to the pendrive
27. Print the document to document image writer and save it to the desktop
28. Close Word and Compress the file
29. Eject the Pendrive
30. Open Paint
31. Draw a quick picture
32. Save it to the desktop
33. Right Click on that picture and zip it to a zip file using "winzip"
34. Open calculator and perform 72 x 36 =, then exit
35. Go to Seacrh, for files and folders, and do a search for all files ending in .doc, then .txt
36. Open Powerpoint and create a blank slide with the word "text" on it and an imported picture from the sample pictures folder, close it without saving

Appendix D: Individual Scenario Results

In this Appendix the individual results of each scenario are explained. In each figure, the item in bold with a shaded box was where the detection of the malicious act occurred.

1. **Scenario 1a:** Attempt to Circumvent Auditing (Clear Audit Logs), as Admin

   Description: This type of scenario is representative of a malicious user attempting to cover his tracks by deleting any detection of the malicious acts that he performed. It is usually one of the last actions a malicious user performs. In this scenario, a user with administrator rights (username: levoy) attempts to delete the audit logs, from within Eventviewer.

   Expected Results: It is expected that a success audit for policy change will be logged. This is expected to be an event 517 in the system events category (S9).

Table A1: Results of Scenario 1a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
1a	**1**	0	**1**	0	0	0	**1**	0	**1**	0	**1**	0	**4**	0	**2**	0	**2**	0	**Yes**

   Results: An event 517 was logged in all of the categories (S1, S2, S3, S4, S5, S6, S7, S8, and S9).[Figure 4.5] It can be determined by the event 517 that the audit log was cleared by user "levoy" at 12:09:13 AM on 4/9/06. The event itself can be found in Appendix A, listed under the scenario number.

   Discussion: This event was easy to determine since it is logged in every category, and there are no other events logged for the scenario in most categories. This would be a very important event to notice, because it is rare for the security audit logs to be deleted under normal circumstances and it is very frequently used by a malicious to cover tracks. It may not show any information about what occurred prior to the deletion of the audit logs, but it is worth looking onto the reason that it was performed.

2. **Scenario 2a:** Attempt to access another users folder as admin (Users Folder has no *SACL*, but file inside does. Admin has access to folder and file)

   Description: This type of scenario is representative of a possible malicious user attempting to browse to another users "My Documents" folder on a shared workstation. While this may simply be a case of a user being curious, it could be a malicious user trying to gain access to another users important information. In this scenario, it is being performed by user *levoy* ,who has administrator rights, tries to access *`C:\Documents and Settings\User2\My Documents\User 2.txt`*. By default, Windows XP® limits access to users files located in their individual "`C:\Documents and Settings\UserXX\My Documents`" folder to the individual user and the administrator group, and no auditing is enabled for successful or failed attempts to access these files. For this scenario an *SACL* was configured on the file "`User 2.txt`" to allow access by all users and log successful and failed attempts.

   Expected Results: It is expected that a success audit for object access will be logged. This is expected to be an event 560 in the object access category (S5).

Table A.2: Results of Scenario 2a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
2a	0	0	0	0	0	0	0	0	15	0	0	0	0	0	2	0	0	0	Yes

   Results: A success audit for object access was logged as event 560. It can be determined from the log that user "levoy" accessed "C:\Documents and Settings\UserX\My Documents\User 2.txt" on 3/25/2006 at 09:47:39 hours. The event itself can be found in Appendix A, listed under the scenario number.

   Discussion: This event determines that the user "levoy" gained access to user 2's file, and that permission was granted because it was logged as a successful attempt and was logged as a success audit. The access to user 2's "Documents and Settings\My Documents" folder was not logged because it was not enabled with auditing through an *SACL*, while access to the file within it called "User 2.txt was

audited because it was enabled with an *SACL* for auditing. This audit event would not usually show up because auditing for user files located in their individual "Documents and Settings\My Documents " folder is not enabled by default. In this case it was enabled for auditing, and the logged access could mean that either the user "levoy" simply had specific permission in the *SACL* to view the file, or that they had permission because of being in a group like the administrator group. In an instance like this, the security administrators of an organization need to look deeper into the access to determine if it was authorized by the security policy because it could be a completely normal occurrence of or a malicious act.

3. **Scenario 3a:** Attempt to access a users file successfully as admin with *SACL* (Admin has access to file, auditing is enabled in *SACL*)

   Description: This type of scenario is representative of a possible malicious user attempting to browse to another users file, which has been configured with a *SACL*, located on a shared workstation. The *SACL* controls access to the file as well as auditing all successful and unsuccessful attempts to access the file. In this scenario, the user "levoy" who has administrator rights attempts to access "C:\User 2 (with Admin and User 4 Access)\User 2.txt" , which has User 2 as an owner and is configured to allow access to user levoy, and to log all successful and failed attempts to access it.

   Expected Results: It is expected that a success audit for object access will be logged. This is expected to be an event 560 in the object access category (S5).

Table A.3: Results of Scenario 3a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
3a	0	0	0	0	0	0	0	0	62	0	0	0	0	0	2	0	0	0	Yes

   Results: A success audit for object access was logged as event 560. It can be determined from the log that user "levoy" accessed "C:\User 2 (with Admin and User 4 Access)\User 2.txt" on 3/25/2006 at 09:47:52 hours. The event itself can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" gained access to user 2's file, and that they had permission to do so because it was a successful attempt and was logged as a success audit. This could mean that either the user "levoy" simply had specific permission in the *SACL* to view the file, or that they had permission because of being in a group like the administrator group. In an instance like this, the security administrators of an organization need to look deeper into the event to determine if it the users access to the file was authorized by the security policy.

4. **Scenario 4a:** Attempt to access a users file unsuccessfully as admin with *SACL* (Admin Does NOT have access to file, auditing is enabled in *SACL*)

   Description: This scenario is the same as scenario 3a except the *SACL* does not allow the user *levoy* access to the file. In this scenario, the user *levoy* who has administrator access attempts to access "`C:\User 2 (Only User 2 Access)\User 2.txt`" in which *User 2* is the owner and the file is configured without access to the user *levoy* and to log all successful and failed attempts to access it.

   Expected Results: It is expected that a failure audit for object access will be logged. This is expected to be an event 560 in the object access category (F5).

Table A4.: Results of Scenario 4a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
4a	0	0	0	0	0	0	0	0	0	**108**	0	0	0	0	0	0	0	0	**Yes**

Results: A failure audit for object access was logged as event 560. It can be determined from the log that user "levoy" attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 09:49:47 hours. The event itself can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" attempted to gain access to user 2's file, and that he/she did not have permission to do so because it was a failed attempt that was logged as a failure audit. In an instance like this, the security administrators of an organization need to look deeper into the event to determine

why the user attempted to access an unauthorized file and if the event was an instance of a simple mistake or a malicious attempt to gain access to an unauthorized file.

5. **Scenario 5a:** Attempt to access a users file as admin with *SACL* by changing users password and logging in as the user.

   Description: This type of scenario is representative of a possible malicious insider with administrator rights attempting to gain access to a users file in which he doesn't have access by changing the users password. In this scenario, the user *levoy* who has administrator access desires to gain access to "`C:\User 2 (Only User 2 Access)\User 2.txt`" and does not have access, so he/she changes *User 2*'s password and then logs on as User 2 to gain access to the file.

   Expected Results: It is expected that a success audit for account management will be logged. This is expected to be an event 628 in the account management category (S2) showing that an account password was changed.

Table A.5: Results of Scenario 5a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
5a	1	7	2	0	0	0	6	6	86	0	0	0	1	0	35	0	0	0	Yes

   Results: A success audit for account management was logged as event 628. It can be determined from the log that user "levoy" changed the password for the "User 2" acount on 3/25/2006 at 7:44:01 PM. Two secondary events that can be used to determine the entire evolution of events are event 528 that shows that after the password was changed, the "User 2" account was logged onto, and event 560 that shows that the file "C:\User 2 (Only User 2 Access)\User 2.txt" was then accessed by User 2.

   Discussion: This event determines that the user "levoy" attempted to gain access to user 2's file by changing the password to the account, and then logged onto it and gained access to it's files. While this approach to gaining access to a users files is

less likely than taking ownership of the files(scenario 14a), it is still a possibility. The organization security policy should specify a way of accounting for administrator actions such as resetting a password, so that legitimate administrator duties can be determined from potential malicious actions.

6. **Scenario 6a**: Attempt to access a users file as admin by creating an admin account to use to mask attempts

   Description: This type of scenario is representative of a possible malicious insider with administrator rights attempting to gain access to a user's file in which he doesn't have access. He/she will attempt to create an account with administrator privileges to mask attempts to gain access to the user's file. In this scenario, the user *levoy* who has administrator rights desires to gain access "`C:\Documents and Settings\User 2\My Documents\User 2.txt`" and does not have access, so he/she creates an account with administrator rights with the username *admin*, and then logs on to that account to gain access to the file.

   Expected Results: It is expected that a success audit for account management will be logged. This is expected to be an event 628 in the account management category (S2) showing that an account was created.

Table A.6: Results of Scenario 6a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
6a	3	7	6	0	0	0	11	7	21	0	0	0	11	0	88	0	0	0	Yes

   Results: A success audit for account management was logged as event 628. It can be determined from the log that user "levoy" created an account named "admin" on 3/25/2006 at 7:46:15 PM. A secondary event that can be used to determine the entire evolution of events is event 528 which is the successful logon of the account "admin", followed by the event 560 which is the successful access to the file "C:\Documents and Settings\User 2\My Documents\User 2.txt" by the account "admin".

Discussion: This event determines that the user "levoy" attempted to gain access to user 2's file and mask the effort by creating another administrator account and using it to gain access to the file, in an effort to divert attention away from its own account. This would not be a very complicated or likely scenario but still could happen and in this case it was logged and relatively easily identified. Just as all other administrator duties, the organization security policy should specify a way of accounting for administrator actions such as creating or deleting accounts, so that legitimate administrator duties can be determined and separated from potential malicious actions.

7. **Scenario 7a**: Attempt to access a users file as admin by changing a user account to admin to use to mask attempts

   Description: This type of scenario is representative of a possible malicious insider with administrator rights attempting to gain access to a user's file in which he doesn't have access. He/she will attempt to change a normal user account to an administrator account to mask attempts to gain access to the user's file. In this scenario, the user levoy who has administrator access desires to gain access "C:\Documents and Settings\User 2\My Documents\User 2.txt" and does not have access, so he/she changes the account User 1, which is a normal user account, to have administrator rights and then logs onto that account to gain access to the file.

   Expected Results: It is expected that a success audit for account management will be logged. This is expected to be an event 636 in the account management category (S2) showing that the account "User 1" was changed to have administrator rights.

Table A.7: Results of Scenario 7a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
7a	2	8	**1**	0	0	0	7	8	33	0	0	0	10	0	47	0	0	0	**Yes**

Results: A success audit for account management was logged as event 636. It can be determined from the log that user "levoy" changed account with ID: S-1-5-21-484763869-152049171-839522115-1004 (which can be determined to be User 1

from event 528, see appendix A) to have administrator privileges on 3/25/2006 at 7:55:08 PM. A secondary event that can be used to determine the entire evolution of events is event 528 which is the successful logon of the account "User 1", followed by the event 560 which is the successful access to the file "C:\Documents and Settings\User 2\My Documents\User 2.txt" by the account "User 1".

Discussion: This event determines that the user "levoy" attempted to gain access to user 2's file and mask the effort by changing a user account to an administrator account and using it to gain access to the file, in an effort to divert attention away from its own account. This would not be a very complicated or likely scenario but still could happen and in this case it was logged and relatively easily identified. Just as all other administrator duties, the organization security policy should specify a way of accounting for administrator actions such as creating or deleting accounts, so that legitimate administrator duties can be determined and separated from potential malicious actions.

8. **Scenario 8a:** Attempt to access a users file by changing a user account to admin by placing into admin group to mask attempts to access users file

   Description: This type of scenario is representative of a possible malicious insider with administrator rights attempting to gain access to a user's file in which he doesn't have access. He/she will attempt to change a normal user account to an administrator account to mask attempts to gain access to the user's file. In this scenario, the user *levoy* who has administrator access desires to gain access to "`C:\Documents and Settings\User 2\My Documents\User 2.txt`" and does not have access, so he/she changes the account *User 3*, which is a normal user account, to have administrator rights by adding it to the administrator group and then logs onto that account to gain access to the file.

   Expected Results: It is expected that a success audit for account management will be logged. This is expected to be an event 636 in the account management category

(S2) showing that the account “User 3” was changed to have administrator rights by being added to the administrator group.

Table A.8: Results of Scenario 8a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
8a	0	1	1	0	0	0	9	2	17	0	0	0	10	0	46	0	0	0	Yes

Results: A success audit for account management was logged as event 636. It can be determined from the log that user “levoy” added account with ID: S-1-5-21-484763869-152049171-839522115-1006 (which can be determined to be User 3 from event 528, see appendix A) to the administrator group on 3/25/2006 at 7:57:18 PM. A secondary event that can be used to determine the entire evolution of events is event 528 which is the successful logon of the account “User 3”, followed by the event 560 which is the successful access to the file “C:\Documents and Settings\User 2\My Documents\User 2.txt” by the account “User 3”.

Discussion: This event determines that the user “levoy” attempted to gain access to user 2’s file and mask the effort by changing a user account to an administrator account and using it to gain access to the file, in an effort to divert attention away from its own account. This would not be a very complicated or likely scenario but still could happen and in this case it was logged and relatively easily identified. Just as all other administrator duties, the organization security policy should specify a way of accounting for administrator actions such as creating or deleting accounts, so that legitimate administrator duties can be determined and separated from potential malicious actions. It is also very important for security administrators to pay close attention to the members of the administrator group.

9. **Scenario 9a**: Attempt to access a users file as admin by attempting to create a hardlink

Description: This type of scenario is representative of a possible malicious insider with administrator rights attempting mask attempts to access a users file by creating a hard link to a users file from the command prompt. A hardlink is a link that

points to a particular file or directory and has all the same rights as the file itself. In this scenario, the user levoy who has administrator rights desires to gain access to “C:\Documents and Settings\User 2\My Documents\User 2.txt”. He/she goes to the command prompt and types "fsutil hardlink create test "C:\Documents and Settings\User 2\My Documents\User 2.txt" which will create a hardlink to the file "C:\Documents and Settings\User 2\My Documents\User 2.txt" named “test”.

Expected Results: It is expected that a success audit for object access will be logged. This is expected to be an event 568 in the object access category (S5) showing that a successful attempt to make a hard link was performed.

Table A.9: Results of Scenario 9a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
9a	3	2	0	0	0	0	2	1	**6**	0	0	0	5	0	12	0	0	0	**Yes**

Results: A success audit for object access was logged as event 568. It can be determined from the log that user “levoy” successfully created a hard link to the file on 3/25/2006 at 9:59:58 PM.

Discussion: This event determines that the user “levoy” successfully attempted to gain access to user 2’s file by creating a hardlink. This is common way that malicious insiders gain access to files, and then delete the hardlink because if the file has an *SACL* enabled, it will log this event but it will be less noticeable than simple access to the file.[add ref]. Event 568 is an event that security administrators need to be aware of and look for on a regular basis because it is seldom used under normal circumstances.

10. **Scenario 10a**: Attempt to access a users file as admin with an *SACL* by attempting to create a hardlink

Description: This type of scenario is representative of a malicious insider with administrator rights attempting to gain access to a user’s file in which he does not

have access by attempting to create a hard link to the users file from the command prompt. . The file has an *SACL* restricting access to only User 2. In this scenario, the user levoy who has administrator access desires to gain access to the file “C:\User 2 (Only User 2 Access)\User 2.txt" and does not have access. He/she goes to the command prompt and types " fsutil hardlink create test2 "c:\User 2 (Only User 2 Access)\User 2.txt" which will attempt to create a hardlink to the file "fsutil hardlink create test2 "c:\User 2 (Only User 2 Access)\User 2.txt" named “test2”.

Expected Results: It is expected that a success audit for object access will be logged. This is expected to be an event 568 in the object access category (S5) showing that a successful attempt to make a hard link was performed.

Table A.10: Results of Scenario 10a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
10a	0	1	0	0	0	0	2	0	0	**1**	0	0	0	0	12	0	0	0	**Yes**

Results: A failure audit for object access was logged as event 560. It can be determined from the log that user “levoy” unsuccessfully attempted to created a hard link to the file on 3/25/2006 at 10:00:31 PM.

Discussion: This event determines that the user “levoy” unsuccessfully attempted to gain access to user 2’s file by creating a hardlink. This was an unexpected result. Instead of an event 568 failure hardlink attempt, an event 560 failure object access was logged. Just as scenario 9a, this is common way that malicious insiders gain access to files, and then delete the hardlink because if the file has an *SACL* enabled, it will log this event but it will be less noticeable than simple access to the file.[add ref]. Security administrators need to regularly review audit logs and look for event 560 failure object access to determine if they are simple mistakes or if a possible malicious insider is browsing another users files.

11. Scenario 11u: Attempt to access another users folder as user (Users Folder has no *SACL*, but file inside does have *SACL*)

Description: This type of scenario is representative of a possible malicious insider with normal user rights attempting to gain access to another user's file located in their personal "My Documents" folder on a shared workstation. It would be indicative of a user either being curious and browsing files on a workstation, or trying to gain access to personal information for malicious reasons. He/she should not be able to browse to this folder because normal users are not allowed into other users folders by default. In this scenario, the user User 4 who has normal user access desires to gain access to "C:\Documents and Settings\User 2\My Documents\User 2.txt". He/she simply tries to browse to it.

Expected Results: It is expected that a failure audit for object access will be logged. This is expected to be an event 560 in the object access category (F5) showing that a failed object access attempt was performed.

Table A.11: Results of Scenario 11u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
11u	1	0	0	0	0	0	2	1	0	0	0	0	1	0	10	0	0	0	No

Results: There was nothing logged for this scenario to indicate what occurred.

Discussion: Since normal user access to other users "C:\Documents and Settings\User XX\My Documents" folder is not allowed by default on Windows XP®, the user in this scenario was blocked from gaining access to this folder. Also by default there is nothing logged for attempts to gain access, so there was no evidence of the attempt. Even though the file "User 2.txt" itself was enabled for auditing, the folder "My Documents" was not. If an organizations security administrators want to know if there are failed or successful attempts to gain access

to other users files then they need to enable auditing on those files and have an *SACL* on them as well.

12. **Scenario 12u**: Attempt to access another users file as user successfully with *SACL* (User has access to file, Auditing is enabled in *SACL*)

Description: This scenario is representative of a possible malicious user with normal user rights attempting to browse to another users file on a shared workstation. The file has been configured with an *SACL*, which controls access to the file as well as auditing all successful and unsuccessful attempts to access the file. In this scenario, the user "User 4" who has normal user rights attempts to access the file "C:\User 2 (with Admin and User 4 Access)\User 2.txt" which User 2 is an owner. The file is configured to allow access to user "User 4", and to log all successful and failed attempts to access it.

Expected Results: It is expected that a success audit for object access will be logged. This is expected to be an event 560 in the object access category (S5).

Table A.12: Results of Scenario 12u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
12u	0	0	0	0	0	0	0	0	54	0	0	0	0	0	3	0	0	0	Yes

Results: A success audit for object access was logged as event 560. It can be determined from the log that user "User 4" accessed "C:\User 2 (with Admin and User 4 Access)\User 2.txt" on 3/25/2006 at 10:01:52 hours. The event itself can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "User 4" gained access to user 2's file, and that they had permission to do so because it was a successful attempt and was logged as a success audit. This could mean that either the user "User 4" simply had specific permission in the *SACL* to view the file, or that they had permission because of being in a group like the administrator group. In an instance like this,

the security administrators of an organization need to look deeper into the event to determine if it the users access to the file was authorized by the security policy.

13. **Scenario 13u**: Attempt to access another users file as user unsuccessfully with *SACL* (User does not have access to file, Auditing is enabled in *SACL*)

Description: This type of scenario is representative of a malicious user with normal user rights attempting to browse to another users file on a shared workstation. The file has been configured with an *SACL*, which controls access to the file as well as auditing all successful and unsuccessful attempts to access the file. In this scenario, the user "User 4" who has normal user rights attempts to access the file "C:\User 2 (Only User 2 Access)\User 2.txt", in which User 2 is the owner and the file is configured to not allow access to "User 4", and to log all successful and failed attempts to access it.

Expected Results: It is expected that a failure audit for object access will be logged. This is expected to be an event 560 in the object access category (F5).

Table A.13: Results of Scenario 13u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
13u	0	0	0	0	0	0	2	0	0	108	0	0	0	0	9	0	0	0	Yes

Results: A failure audit for object access was logged as event 560. It can be determined from the log that user "User 4" attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 10:01:57 hours. The event itself can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "User 4" attempted to gain access to user 2's file, and that he/she did not have permission to do so because it was a failed attempt that was logged as a failure audit. In an instance like this, the

security administrators of an organization need to look deeper into the event to determine why the user attempted to access an unauthorized file and if the event was an instance of a simple mistake or a malicious attempt to gain access to an unauthorized file.

14. **Scenario 14a**: Attempt to access a users file as Admin with *SACL* by changing ownership of the file

Description: This type of scenario is representative of a possible malicious user with administrator rights attempting to gain access to another users file on a shared workstation. The file has been configured with an *SACL*, which controls access to the file as well as auditing all successful and unsuccessful attempts to access the file. In this scenario, the user "levoy" who has administrator rights attempts to access the file "C:\User 2 (Only User 2 Access)\User 2.txt", in which User 2 is the owner and it is configured without access to "levoy" and to log all successful and failed attempts to access it. The user "levoy" attempts to gain access by taking ownership of the file and giving himself access to it.

Expected Results: It is expected that most likely initially a failure audit for object access will be logged as user "levoy" tries to access the file as an event 560 in the object access category (F5). Then it is expected that a success audit for object access will be logged as an event 560 with the take ownership privilege (SeTakeOwnershipPrivilege) indicated in the privileges section.

Table A.14: Results of Scenario 14a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
14a	0	1	0	0	0	0	4	1	**76**	78	0	0	13	0	22	0	0	0	**Yes**

Results: The primary audit log indication was a success audit indicating object access with the take ownership privilege (SeTakeOwnershipPrivilege) used in the object access category (S5). Secondary audit log indications show that a failure audit for object access was logged as event 560 prior to the take ownership privilege being exercised in the object access category (F5). Another secondary

audit log indication was that the take ownership privilege (SeTakeOwnershipPrivilege) was exercised was logged as event 578 in the privilege use category. It can be determined from the logs that "levoy" unsuccessfully attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 10:02:42 hours, then used the take ownership privilege to take ownership of the file and accessed it on 3/25/2006 at 10:02:42 hours. The events can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" attempted unsuccessfully to gain access to user 2's file first, and then took ownership of the file and gave himself access. This scenario requires close evaluation of the event 560 audit logs to determine that the take ownership privilege was used. If the confidentiality of particular files on workstations is a concern of the organization, then *SACL*'s should be put on those files and auditing should be enabled for success in the category object access. Potential abuse of administrator privileges such as what occurred in this scenario is why auditing should be used to keep an eye on administrators to ensure that they are abiding by the organizations security policy.

15. **Scenario 15a:** Attempt to access a users file as admin by attempting to change the Security Policy

Description: This type of scenario is representative of a possible malicious user with administrator rights disabling all auditing by changing the local security policy to mask attempts to gain access to another users file on a shared workstation. The file has been configured with an *SACL*, which controls access to the file as well as auditing all successful and unsuccessful attempts to access the file. In this scenario, the user "levoy" who has administrator rights disables auditing to mask his/her attempts to access the file "C:\User 2 (With Admin and User 4)\User 2.txt", in which User 2 is the owner and it is configured to allow access to "levoy" and to log all successful and failed attempts to access it.

Expected Results: It is expected that a success audit for object access will be logged as user "levoy" tries to access the file as an event 560 in the object access

category (F5). Then it is expected that a success audit for object access will be logged as an event 560 with the take ownership privilege (SeTakeOwnershipPrivilege) indicated in the privileges section.

Table A.15: Results of Scenario 15a

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
15a	2	1	1	0	0	0	3	1	5	0	1	0	6	0	13	0	1	0	Yes

Results: This scenario is representative of a possible malicious user with normal user rights attempting to elevate privileges to administrator by rebooting the computer into safe mode and trying to guess the password to the administrator account. The administrator account, if not removed or renamed, is created at the time of installing Windows XP®. It is a way of logging into the workstation in safe mode, and there is no limit to the number of failed attempts to try to log into it. In this scenario, a user will reboot the virtual machine and make three failed attempts to guess the administrator password before getting it right and logging in.

Discussion: This event determines that the user "levoy" attempted unsuccessfully to gain access to user 2's file first, and then took ownership of the file and gave himself access. This scenario requires close evaluation of the event 560 audit logs to determine that the take ownership privilege was used. If the confidentiality of particular files on workstations is a concern of the organization, then *SACL*'s should be put on those files and auditing should be enabled for success in the category object access. Potential abuse of administrator privileges such as what occurred in this scenario is why auditing should be used to keep an eye on administrators to ensure that they are abiding by the organizations security policy.

16. **Scenario 16a**: Attempt to guess Admin password by rebooting into safemode and guessing admin password

Description: This scenario is representative of a possible malicious user with normal user rights attempting to elevate privileges to administrator by rebooting the computer into safe mode and trying to guess the password to the administrator

account. The administrator account, if not removed or renamed, is created at the time of installing Windows XP®. It is a way of logging into the workstation in safe mode, and there is no limit to the number of failed attempts to try to log into it. In this scenario, a user will reboot the virtual machine and make three failed attempts to guess the administrator password before getting it right and logging in.

Expected Results: It is expected that a success audit for object access will be logged as user "levoy" tries to access the file as an event 560 in the object access category (F5). Then it is expected that a success audit for object access will be logged as an event 560 with the take ownership privilege (SeTakeOwnershipPrivilege) indicated in the privileges section.

Table A.16: Results of Scenario 16u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
16u	**1**	4	0	0	0	0	**9**	4	12	3	1	0	39	0	40	0	28	0	**Yes**

Results: The primary audit log indication was a success audit indicating object access with the take ownership privilege (SeTakeOwnershipPrivilege) used in the object access category (S5). Secondary audit log indications show that a failure audit for object access was logged as event 560 prior to the take ownership privilege being exercised in the object access category (F5). Another secondary audit log indication was that the take ownership privilege (SeTakeOwnershipPrivilege) was exercised was logged as event 578 in the privilege use category. It can be determined from the logs that "levoy" unsuccessfully attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 10:02:42 hours, then used the take ownership privilege to take ownership of the file and accessed it on 3/25/2006 at 10:02:42 hours. The events can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" attempted unsuccessfully to gain access to user 2's file first, and then took ownership of the file and gave himself access. This scenario requires close evaluation of the event 560 audit logs

to determine that the take ownership privilege was used. If the confidentiality of particular files on workstations is a concern of the organization, then *SACL*'s should be put on those files and auditing should be enabled for success in the category object access. Potential abuse of administrator privileges such as what occurred in this scenario is why auditing should be used to keep an eye on administrators to ensure that they are abiding by the organizations security policy.

17. **Scenario 17u**: Access a users file by booting with a bootable cd and deleting password

Description: This scenario is representative of a possible malicious user with normal user rights attempting to gain access to a users file by rebooting the computer using a bootable cd, changing the users password, then rebooting and gaining access to the users file with the new password. The cd contains a Windows® registry editor and boots into a Linux shell. In this scenario, a user will reboot the virtual machine with the bootable cd inserted. The user will then use the Windows® registry editor on the cd to change the "User 2" account to have no password. Then the user will reboot again, log into the "User 2" account, and gain access to the "C:\User 2 (Only User 2 Access)" file.

Expected Results: It is expected that a success audit for object access will be logged as user "levoy" tries to access the file as an event 560 in the object access category (F5). Then it is expected that a success audit for object access will be logged as an event 560 with the take ownership privilege (SeTakeOwnershipPrivilege) indicated in the privileges section.

Table A.17: Results of Scenario 17u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
17u	2	0	0	0	0	0	35	0	73	0	12	0	22	0	36	7	**16**	0	**Yes**

Results: The primary audit log indication was a success audit indicating object access with the take ownership privilege (SeTakeOwnershipPrivilege) used in the object access category (S5). Secondary audit log indications show that a failure

audit for object access was logged as event 560 prior to the take ownership privilege being exercised in the object access category (F5). Another secondary audit log indication was that the take ownership privilege (SeTakeOwnershipPrivilege) was exercised was logged as event 578 in the privilege use category. It can be determined from the logs that "levoy" unsuccessfully attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 10:02:42 hours, then used the take ownership privilege to take ownership of the file and accessed it on 3/25/2006 at 10:02:42 hours. The events can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" attempted unsuccessfully to gain access to user 2's file first, and then took ownership of the file and gave himself access. This scenario requires close evaluation of the event 560 audit logs to determine that the take ownership privilege was used. If the confidentiality of particular files on workstations is a concern of the organization, then *SACL*'s should be put on those files and auditing should be enabled for success in the category object access. Potential abuse of administrator privileges such as what occurred in this scenario is why auditing should be used to keep an eye on administrators to ensure that they are abiding by the organizations security policy.

18. **Scenario 18u**: Run a rootkit

Description: This scenario is representative of a possible malicious user with normal user rights attempting to elevate privileges by running a rootkit. In this scenario, user "levoy" will insert a pen drive with a rootkit executable file on it. The rootkit is called "Fu_Rootkit", and is a common way of inserting a backdoor into a workstation to enable enumeration at a later time. The user "levoy" will then run the executable "fu.exe" which will install the rootkit, eject the pen drive, and log off.

Expected Results: It is expected that a success audit for object access will be logged as user "levoy" tries to access the file as an event 560 in the object access category (F5). Then it is expected that a success audit for object access will be

logged as an event 560 with the take ownership privilege (SeTakeOwnershipPrivilege) indicated in the privileges section.

Table A.18: Results of Scenario 18u

Scenario	S1	F1	S2	F2	S3	F3	S4	F4	S5	F5	S6	F6	S7	F7	S8	F8	S9	F9	Detected
18u	1	1	0	0	0	0	2	1	0	0	11	0	8	0	30	0	0	0	Yes

Results: The primary audit log indication was a success audit indicating object access with the take ownership privilege (SeTakeOwnershipPrivilege) used in the object access category (S5). Secondary audit log indications show that a failure audit for object access was logged as event 560 prior to the take ownership privilege being exercised in the object access category (F5). Another secondary audit log indication was that the take ownership privilege (SeTakeOwnershipPrivilege) was exercised was logged as event 578 in the privilege use category. It can be determined from the logs that "levoy" unsuccessfully attempted to access "C:\User 2 (Only User 2 Access)\User 2.txt" on 3/25/2006 at 10:02:42 hours, then used the take ownership privilege to take ownership of the file and accessed it on 3/25/2006 at 10:02:42 hours. The events can be found in Appendix A, listed under the scenario number.

Discussion: This event determines that the user "levoy" attempted unsuccessfully to gain access to user 2's file first, and then took ownership of the file and gave himself access. This scenario requires close evaluation of the event 560 audit logs to determine that the take ownership privilege was used. If the confidentiality of particular files on workstations is a concern of the organization, then *SACL's* should be put on those files and auditing should be enabled for success in the category object access. Potential abuse of administrator privileges such as what occurred in this scenario is why auditing should be used to keep an eye on administrators to ensure that they are abiding by the organizations security policy.

Appendix E: *CEM* Method Output from the Program Written in C++

The Configuration Evaluation Method (*CEM*) is used to create an optimal audit configuration based on an exhaustive comparison of all possible audit configurations, using a program written in C++ for this experiment. The program's output can be seen below:

1. The output using weights of W_1=1, W_2=0, and W_3=0:

```
Reading cost.csv file......done.
Reading primary.csv file......done.
Reading secondary.csv file......done.

W1 = 1.0000 W2 = 0.0000 W3 = 0.0000

Per Switch Calculations:
Switch:  0 DC: 0.1667 AC:   177 FPC:     0 AC/TC: 0.0094 FPC/AC: 0.0000
Switch:  1 DC: 0.0000 AC:    33 FPC:     0 AC/TC: 0.0018 FPC/AC: 0.0000
Switch:  2 DC: 0.3333 AC:    12 FPC:     2 AC/TC: 0.0006 FPC/AC: 0.1667
Switch:  3 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch:  4 DC: 0.0000 AC:   480 FPC:     0 AC/TC: 0.0255 FPC/AC: 0.0000
Switch:  5 DC: 0.0000 AC:   160 FPC:     0 AC/TC: 0.0085 FPC/AC: 0.0000
Switch:  6 DC: 0.1667 AC:   575 FPC:   350 AC/TC: 0.0305 FPC/AC: 0.6087
Switch:  7 DC: 0.0000 AC:   192 FPC:   188 AC/TC: 0.0102 FPC/AC: 0.9792
Switch:  8 DC: 0.3889 AC:   941 FPC:   187 AC/TC: 0.0499 FPC/AC: 0.1987
Switch:  9 DC: 0.1667 AC:  1418 FPC:  1130 AC/TC: 0.0752 FPC/AC: 0.7969
Switch: 10 DC: 0.1111 AC:    26 FPC:     0 AC/TC: 0.0014 FPC/AC: 0.0000
Switch: 11 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 12 DC: 0.1111 AC:   290 FPC:   167 AC/TC: 0.0154 FPC/AC: 0.5759
Switch: 13 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 14 DC: 0.1667 AC: 14169 FPC:  6296 AC/TC: 0.7518 FPC/AC: 0.4444
Switch: 15 DC: 0.0000 AC:   327 FPC:     0 AC/TC: 0.0174 FPC/AC: 0.0000
Switch: 16 DC: 0.1667 AC:    47 FPC:     0 AC/TC: 0.0025 FPC/AC: 0.0000
Switch: 17 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000

NIST EC  SwitchID:28882
DC: 0.444 AC: 837 AC/TC: 0.044 FPC: 352 FPC/AC: 0.421 TC: 18847 FOM:
0.444444
Switches on: S1 S2 S4 S6 S9
Total Switches On: 5

NIST SSLF SwitchID:3CD92
DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM:
0.611111
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F7 S9
Total Switches On: 10

SNAC      SwitchID:3CDD3
```

```
DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM:
0.611111
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F6 F7 S9 F9
Total Switches On: 12

Top Ten Switches based upon W1=1.000000 W2=0.000000 W3=0.000000:

Switch Rank:  1 SwitchID:08B0A
DC: 0.944 AC: 17162 AC/TC: 0.911 FPC: 7965 FPC/AC: 0.464 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S8 S9
Total Switches On: 6

Switch Rank:  2 SwitchID:08B0B
DC: 0.944 AC: 17162 AC/TC: 0.911 FPC: 7965 FPC/AC: 0.464 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S8 S9 F9
Total Switches On: 7

Switch Rank:  3 SwitchID:08B0E
DC: 0.944 AC: 17489 AC/TC: 0.928 FPC: 7965 FPC/AC: 0.455 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S8 F8 S9
Total Switches On: 7

Switch Rank:  4 SwitchID:08B0F
DC: 0.944 AC: 17489 AC/TC: 0.928 FPC: 7965 FPC/AC: 0.455 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S8 F8 S9 F9
Total Switches On: 8

Switch Rank:  5 SwitchID:08B1A
DC: 0.944 AC: 17162 AC/TC: 0.911 FPC: 7965 FPC/AC: 0.464 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 F7 S8 S9
Total Switches On: 7

Switch Rank:  6 SwitchID:08B1B
DC: 0.944 AC: 17162 AC/TC: 0.911 FPC: 7965 FPC/AC: 0.464 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 F7 S8 S9 F9
Total Switches On: 8

Switch Rank:  7 SwitchID:08B1E
DC: 0.944 AC: 17489 AC/TC: 0.928 FPC: 7965 FPC/AC: 0.455 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 F7 S8 F8 S9
Total Switches On: 8

Switch Rank:  8 SwitchID:08B1F
DC: 0.944 AC: 17489 AC/TC: 0.928 FPC: 7965 FPC/AC: 0.455 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 F7 S8 F8 S9 F9
Total Switches On: 9
```

```
Switch Rank:  9 SwitchID:08B2A
DC: 0.944 AC: 17452 AC/TC: 0.926 FPC: 8132 FPC/AC: 0.466 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S7 S8 S9
Total Switches On: 7

Switch Rank: 10 SwitchID:08B2B
DC: 0.944 AC: 17452 AC/TC: 0.926 FPC: 8132 FPC/AC: 0.466 TC: 18847 FOM:
0.944444
Switches on: S2 S4 S5 F5 S7 S8 S9 F9
Total Switches On: 8
```

2. The output using weights of W_1=1, W_2=0.07, and W_3=0:

```
Reading cost.csv file......done.
Reading primary.csv file......done.
Reading secondary.csv file......done.

W1 = 1.0000 W2 = 0.0700 W3 = 0.0000

Per Switch Calculations:
Switch:  0 DC: 0.1667 AC:   177 FPC:     0 AC/TC: 0.0094 FPC/AC: 0.0000
Switch:  1 DC: 0.0000 AC:    33 FPC:     0 AC/TC: 0.0018 FPC/AC: 0.0000
Switch:  2 DC: 0.3333 AC:    12 FPC:     2 AC/TC: 0.0006 FPC/AC: 0.1667
Switch:  3 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch:  4 DC: 0.0000 AC:   480 FPC:     0 AC/TC: 0.0255 FPC/AC: 0.0000
Switch:  5 DC: 0.0000 AC:   160 FPC:     0 AC/TC: 0.0085 FPC/AC: 0.0000
Switch:  6 DC: 0.1667 AC:   575 FPC:   350 AC/TC: 0.0305 FPC/AC: 0.6087
Switch:  7 DC: 0.0000 AC:   192 FPC:   188 AC/TC: 0.0102 FPC/AC: 0.9792
Switch:  8 DC: 0.3889 AC:   941 FPC:   187 AC/TC: 0.0499 FPC/AC: 0.1987
Switch:  9 DC: 0.1667 AC:  1418 FPC:  1130 AC/TC: 0.0752 FPC/AC: 0.7969
Switch: 10 DC: 0.1111 AC:    26 FPC:     0 AC/TC: 0.0014 FPC/AC: 0.0000
Switch: 11 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 12 DC: 0.1111 AC:   290 FPC:   167 AC/TC: 0.0154 FPC/AC: 0.5759
Switch: 13 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 14 DC: 0.1667 AC: 14169 FPC:  6296 AC/TC: 0.7518 FPC/AC: 0.4444
Switch: 15 DC: 0.0000 AC:   327 FPC:     0 AC/TC: 0.0174 FPC/AC: 0.0000
Switch: 16 DC: 0.1667 AC:    47 FPC:     0 AC/TC: 0.0025 FPC/AC: 0.0000
Switch: 17 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000

NIST EC  SwitchID:28882
DC: 0.444 AC: 837 AC/TC: 0.044 FPC: 352 FPC/AC: 0.421 TC: 18847 FOM:
0.441336
Switches on: S1 S2 S4 S6 S9
Total Switches On: 5

NIST SSLF SwitchID:3CD92
DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM:
0.601900
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F7 S9
Total Switches On: 10

SNAC      SwitchID:3CDD3
```

DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM: 0.601900
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F6 F7 S9 F9
Total Switches On: 12

Top Ten Switches based upon W1=1.000000 W2=0.070000 W3=0.000000:

Switch Rank: 1 SwitchID:2830A
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 S8 S9
Total Switches On: 6

Switch Rank: 2 SwitchID:2830B
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 S8 S9 F9
Total Switches On: 7

Switch Rank: 3 SwitchID:2831A
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F7 S8 S9
Total Switches On: 7

Switch Rank: 4 SwitchID:2831B
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F7 S8 S9 F9
Total Switches On: 8

Switch Rank: 5 SwitchID:2834A
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F6 S8 S9
Total Switches On: 7

Switch Rank: 6 SwitchID:2834B
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F6 S8 S9 F9
Total Switches On: 8

Switch Rank: 7 SwitchID:2835A
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F6 F7 S8 S9
Total Switches On: 8

Switch Rank: 8 SwitchID:2835B
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM: 0.882181
Switches on: S1 S2 S5 F5 F6 F7 S8 S9 F9
Total Switches On: 9

```
Switch Rank:  9 SwitchID:2C30A
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM:
0.882181
Switches on: S1 S2 F2 S5 F5 S8 S9
Total Switches On: 7

Switch Rank: 10 SwitchID:2C30B
DC: 0.944 AC: 16764 AC/TC: 0.889 FPC: 7615 FPC/AC: 0.454 TC: 18847 FOM:
0.882181
Switches on: S1 S2 F2 S5 F5 S8 S9 F9
Total Switches On: 8
```

3. The output using weights of W_1=1, W_2=0.08, and W_3=0.08:

```
Reading cost.csv file......done.
Reading primary.csv file......done.
Reading secondary.csv file......done.

W1 = 1.0000 W2 = 0.0800 W3 = 0.0800

Per Switch Calculations:
Switch:  0 DC: 0.1667 AC:   177 FPC:     0 AC/TC: 0.0094 FPC/AC: 0.0000
Switch:  1 DC: 0.0000 AC:    33 FPC:     0 AC/TC: 0.0018 FPC/AC: 0.0000
Switch:  2 DC: 0.3333 AC:    12 FPC:     2 AC/TC: 0.0006 FPC/AC: 0.1667
Switch:  3 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch:  4 DC: 0.0000 AC:   480 FPC:     0 AC/TC: 0.0255 FPC/AC: 0.0000
Switch:  5 DC: 0.0000 AC:   160 FPC:     0 AC/TC: 0.0085 FPC/AC: 0.0000
Switch:  6 DC: 0.1667 AC:   575 FPC:   350 AC/TC: 0.0305 FPC/AC: 0.6087
Switch:  7 DC: 0.0000 AC:   192 FPC:   188 AC/TC: 0.0102 FPC/AC: 0.9792
Switch:  8 DC: 0.3889 AC:   941 FPC:   187 AC/TC: 0.0499 FPC/AC: 0.1987
Switch:  9 DC: 0.1667 AC:  1418 FPC:  1130 AC/TC: 0.0752 FPC/AC: 0.7969
Switch: 10 DC: 0.1111 AC:    26 FPC:     0 AC/TC: 0.0014 FPC/AC: 0.0000
Switch: 11 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 12 DC: 0.1111 AC:   290 FPC:   167 AC/TC: 0.0154 FPC/AC: 0.5759
Switch: 13 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000
Switch: 14 DC: 0.1667 AC: 14169 FPC:  6296 AC/TC: 0.7518 FPC/AC: 0.4444
Switch: 15 DC: 0.0000 AC:   327 FPC:     0 AC/TC: 0.0174 FPC/AC: 0.0000
Switch: 16 DC: 0.1667 AC:    47 FPC:     0 AC/TC: 0.0025 FPC/AC: 0.0000
Switch: 17 DC: 0.0000 AC:     0 FPC:     0 AC/TC: 0.0000 FPC/AC: 0.0000

NIST EC  SwitchID:28882
DC: 0.444 AC: 837 AC/TC: 0.044 FPC: 352 FPC/AC: 0.421 TC: 18847 FOM:
0.407248
Switches on: S1 S2 S4 S6 S9
Total Switches On: 5

NIST SSLF SwitchID:3CD92
DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM:
0.546713
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F7 S9
Total Switches On: 10

SNAC      SwitchID:3CDD3
```

```
DC: 0.611 AC: 2480 AC/TC: 0.132 FPC: 1670 FPC/AC: 0.673 TC: 18847 FOM:
0.546713
Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F6 F7 S9 F9
Total Switches On: 12

Top Ten Switches based upon W1=1.000000 W2=0.080000 W3=0.080000:

Switch Rank:  1 SwitchID:3B386
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F8 S9
Total Switches On: 10

Switch Rank:  2 SwitchID:3B387
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F8 S9 F9
Total Switches On: 11

Switch Rank:  3 SwitchID:3B396
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F7 F8 S9
Total Switches On: 11

Switch Rank:  4 SwitchID:3B397
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F7 F8 S9 F9
Total Switches On: 12

Switch Rank:  5 SwitchID:3B3C6
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F6 F8 S9
Total Switches On: 11

Switch Rank:  6 SwitchID:3B3C7
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F6 F8 S9 F9
Total Switches On: 12

Switch Rank:  7 SwitchID:3B3D6
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F6 F7 F8 S9
Total Switches On: 12

Switch Rank:  8 SwitchID:3B3D7
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 S3 F3 S5 F5 S6 F6 F7 F8 S9 F9
Total Switches On: 13
```

```
Switch Rank:  9 SwitchID:3F386
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 F2 S3 F3 S5 F5 S6 F8 S9
Total Switches On: 11

Switch Rank: 10 SwitchID:3F387
DC: 0.889 AC: 3621 AC/TC: 0.192 FPC: 1319 FPC/AC: 0.364 TC: 18847 FOM:
0.844378
Switches on: S1 F1 S2 F2 S3 F3 S5 F5 S6 F8 S9 F9
Total Switches On: 12
```

Appendix F: C++ Code for *CFOM* Program Written for this Experiment

```
#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>

int main(int argc, char* argv[])
{
   char CostLine[18][100];
   char PrimaryLine[18][100];
   char SecondaryLine[18][100];
   char *Value[19];
   char *fp;
   char line[120];
   char *p;
   unsigned int Cost[18][18];
   unsigned int Primary[18][18];
   unsigned int Secondary[18][18];
   unsigned int RowSum[18];
   unsigned int ColSum[18];
   unsigned char i;
   unsigned char j;
   unsigned char detected;
   float sdc[18];
   unsigned int sac[18];
   unsigned int sfpc[18];
   float dc;
   unsigned int ac;
   unsigned int fpc[18];
   unsigned int tfpc;
   unsigned long Top20Switches[21];
   float Top20DC[21];
   unsigned int Top20AC[21];
   unsigned int Top20FPC[21];
   float Top20FOM[21];
   unsigned long SwitchID;
   unsigned long SwitchMask[18];
   unsigned long SwitchIMask[18];
   unsigned int SwitchIndex[18];
   unsigned char Detected[18];
   unsigned int FP[18];
   unsigned int NUCost[18];
   unsigned int NUFP[18];
   unsigned int TC;
   unsigned int AC;
   unsigned int FPC;
   float W1;
```

```
float W2;
float W3;
float ACTC;
float FPCAC;
float FOM;
unsigned int npo;
unsigned int n;
unsigned int found;
unsigned int totalsw;

FILE *costfile;
FILE *primaryfile;
FILE *secondaryfile;

printf("Reading cost.csv file...");
/*costfile=fopen("d:\\cost.csv","r");*/
costfile=fopen("./cost.csv","r");
if(costfile==NULL)
{
   printf("\n Error cannot open file cost.csv\a ");
   exit(0);
}
/* get all 18 lines */
for(i=0; i<18; i++)
{
   fgets(CostLine[i],99,costfile);
}
fclose(costfile);
printf("...done.\n");

for(i=0; i<18; i++)
{
   fp = CostLine[i];
   for(j=0; j<18; j++)
   {
      Value[j]=strtok(fp,",");
      fp+=strlen(fp)+1;
      Cost[i][j]=atoi(Value[j]);
   }
}

printf("Reading primary.csv file...");
/* primaryfile=fopen("d:\\primary.csv","r"); */
primaryfile=fopen("./primary.csv","r");
if(primaryfile==NULL)
{
   printf("\n Error cannot open file primary.csv\a ");
   exit(0);
}
/* get all 18 lines */
for(i=0; i<18; i++)
{
   fgets(PrimaryLine[i],99,primaryfile);
}
fclose(primaryfile);
```

```
printf("...done.\n");

for(i=0; i<18; i++)
{
   fp = PrimaryLine[i];
   for(j=0; j<18; j++)
   {
      Value[j]=strtok(fp,",");
      fp+=strlen(fp)+1;
      Primary[i][j]=atoi(Value[j]);
   }
}

printf("Reading secondary.csv file...");
/* secondaryfile=fopen("d:\\secondary.csv","r"); */
secondaryfile=fopen("./secondary.csv","r");
if(secondaryfile==NULL)
{
   printf("\n Error cannot open file secondary.csv\a ");
   exit(0);
}
/* get all 18 lines */
for(i=0; i<18; i++)
{
   fgets(SecondaryLine[i],99,secondaryfile);
}
fclose(secondaryfile);
printf("...done.\n");

for(i=0; i<18; i++)
{
   fp = SecondaryLine[i];
   for(j=0; j<18; j++)
   {
      Value[j]=strtok(fp,",");
      fp+=strlen(fp)+1;
      Secondary[i][j]=atoi(Value[j]);
   }
}

fprintf(stderr,"Please enter the value for W1: ");
p = gets (line);
if (p == NULL)
{
   printf("ERROR reading user input for W1!\n");
     exit(0);
}
else
{
      W1 = (float)atof(line);
}
/* printf("W1 = %5.4f\n",W1); */

fprintf(stderr,"Please enter the value for W2: ");
p = gets (line);
```

```
if (p == NULL)
{
   printf("ERROR reading user input for W2!\n");
     exit(0);
}
else
{
      W2 = (float)atof(line);
}
/* printf("W2 = %5.4f\n",W2); */

fprintf(stderr,"Please enter the value for W3: ");
p = gets (line);
if (p == NULL)
{
   printf("ERROR reading user input for W3!\n");
     exit(0);
}
else
{
      W3 = (float)atof(line);
}
printf("\nW1 = %5.4f W2 = %5.4f W3 = %5.4f\n",W1,W2,W3);

for(i=0; i<10; i++)
{
   Top20Switches[i]=(unsigned long)0;
   Top20DC[i]=(float)0.0;
     Top20AC[i]=0;
   Top20FPC[i]=0;
   Top20FOM[i]=(float)0.0;
}

/* false positives for given scenarios */
FP[0]=0;
FP[1]=0;
FP[2]=2;
FP[3]=0;
FP[4]=0;
FP[5]=0;
FP[6]=30;
FP[7]=28;
FP[8]=27;
FP[9]=10;
FP[10]=0;
FP[11]=0;
FP[12]=7;
FP[13]=0;
FP[14]=216;
FP[15]=0;
FP[16]=0;
FP[17]=0;

/* normal use cost and false positives for 40 hour week */
NUCost[0]=160;
```

```
NUCost[1]=0;
NUCost[2]=0;
NUCost[3]=0;
NUCost[4]=480;
NUCost[5]=160;
NUCost[6]=480;
NUCost[7]=160;
NUCost[8]=480;
NUCost[9]=1120;
NUCost[10]=0;
NUCost[11]=0;
NUCost[12]=160;
NUCost[13]=0;
NUCost[14]=13760;
NUCost[15]=320;
NUCost[16]=0;
NUCost[17]=0;

NUFP[0]=0;
NUFP[1]=0;
NUFP[2]=0;
NUFP[3]=0;
NUFP[4]=0;
NUFP[5]=0;
NUFP[6]=320;
NUFP[7]=160;
NUFP[8]=160;
NUFP[9]=1120;
NUFP[10]=0;
NUFP[11]=0;
NUFP[12]=160;
NUFP[13]=0;
NUFP[14]=6080;
NUFP[15]=0;
NUFP[16]=0;
NUFP[17]=0;

/*
for(i=0; i<18; i++)
{
   printf("Row %d: ",i);
   for(j=0; j<18; j++)
   {
      printf("%d,",Cost[i][j]);
   }
   printf("\n");
}

for(i=0; i<18; i++)
{
   printf("Row %d: ",i);
   for(j=0; j<18; j++)
   {
      printf("%d,",Primary[i][j]);
   }
```

```
      printf("\n");
   }

   for(i=0; i<18; i++)
   {
      printf("Row %d: ",i);
      for(j=0; j<18; j++)
      {
         printf("%d,",Secondary[i][j]);
      }
      printf("\n");
   }
   */
   for(i=0; i<18; i++)
   {
      RowSum[i]=0;
      ColSum[i]=0;
   }
   TC=0;
   for(i=0; i<18; i++)
   {
      for(j=0; j<18; j++)
      {
         RowSum[i]+=Cost[i][j];
         ColSum[i]+=Cost[j][i];
             TC+=Cost[i][j];
      }
        TC+=NUCost[i];
   }
   /*
   for(i=0; i<18; i++)
   {
      printf("Index: %d RowSum: %d ColSum:
%d\n",i,RowSum[i],ColSum[i]);
   }
   */

   /* per switch (column) calculations */
   printf("\nPer Switch Calculations:\n");
   for(i=0; i<18; i++)
   {
      detected=0;
      sac[i]=0;
      sfpc[i]=0;
      for(j=0; j<18; j++)
      {
         if(Primary[j][i] > 0)
             {
              detected++;
             }
         sac[i]+=Cost[j][i];
      }
      sdc[i]=((float)detected)/((float)18);
      AC=sac[i]+NUCost[i];
        FPC=FP[i]+NUFP[i];
```

```
        if(AC > 0)
        {
         printf("Switch: %2d DC: %5.4f AC: %5d FPC: %5d AC/TC: %5.4f
FPC/AC: %5.4f\n",i,sdc[i],AC,FPC,
                    (float)AC/(float)TC,(float)FPC/(float)AC);
      }
        else
        {
         printf("Switch: %2d DC: %5.4f AC: %5d FPC: %5d AC/TC: %5.4f
FPC/AC: 0.0000\n",i,sdc[i],AC,FPC,
                    (float)AC/(float)TC);
      }
   }

   /* initialize switch mask and index */
   for(i=0; i<18; i++)
   {
      SwitchMask[i] = 0x00001 << i;
      SwitchIMask[i] = 0x20000 >> i;
        SwitchIndex[i] = 17-i;
   }
   /*
   for(i=0; i<18; i++)
   {
      printf("Switch: %d Mask: %05lX IMask: %05lX Index:
%d\n",i,SwitchMask[i],SwitchIMask[i],SwitchIndex[i]);
   }
   */
   for(i=0; i<20; i++)
   {
      Top20Switches[i]=(unsigned long)0;
      Top20DC[i]=(float)0.0;
      Top20AC[i]=0;
      Top20FPC[i]=0;
      Top20FOM[i]=(float)-100.0;
   }

   /*for(SwitchID=0; SwitchID<262144; SwitchID++)*/
   for(SwitchID=0; SwitchID<262144; SwitchID++)
   {
      /* for each unique switch setting */
      /* identify all switches that are on */
      /* clear detection information */
      for(i=0; i<18; i++)
      {
         Detected[i]=0;
      }
      ac=0;
      tfpc=0;

      for(i=0; i<18; i++)
      {
         /* consider all switches */
         if(SwitchIMask[i] & SwitchID)
             {
```

```
              /* switch is on, so see what it contributes */
            /* switch 0 = column 18 */
            /* switch 1 = column 17 */
            /* switch 17 = column 1 */
              /* printf("SwitchID %05lX has Switch %d
set\n",SwitchID,i); */
            for(j=0; j<18; j++)
                  {
                 /* walk rows j for switch i */
               if(Primary[j][i] > 0)
                     {
                      Detected[j]=1;
                       }
               ac+=Cost[j][i];
            }
                  ac+=NUCost[i];
            tfpc+=(FP[i]+NUFP[i]);
         }
      }
      detected=0;

      for(i=0;i<18;i++)
      {
             fpc[i]=0;
             /* walk rows */
         if(Detected[i])
             {
                  /* this row is detected */
              detected++;
             }
        }
      dc=((float)detected)/((float)18);
        ACTC = (float)ac/(float)TC;
      FPCAC = (float)tfpc/(float)ac;
        FOM = (W1*dc) - (W2*ACTC) - (W3*FPCAC);

      /* printf("SwitchID:%05lX DC: %4.3f AC: %d FPC: %d TC: %d FOM:
%8.6f\n",SwitchID,dc,ac,tfpc,TC,FOM);*/

      found=0;
      for(i=0; (i<20)&&(!found); i++)
      {
         if(FOM > Top20FOM[i])
         {
         /*printf("i: %d SwitchID:%05lX DC: %4.3f AC: %d FPC: %d TC: %d
FOM: %8.6f is better than\nSwitchID:%05lX DC: %4.3f AC: %d FPC: %d TC:
%d FOM:
%8.6f\n",i,SwitchID,dc,ac,tfpc,TC,FOM,Top20Switches[0],Top20DC[0],Top20
AC[0],Top20FPC[0],TC,Top20FOM[0]);*/
            for(j=0; j<(20-i); j++)
          {
             npo=20-j;
               n=19-j;
               Top20Switches[npo]=Top20Switches[n];
               Top20DC[npo]=Top20DC[n];
```

```
                Top20AC[npo]=Top20AC[n];
                Top20FPC[npo]=Top20FPC[n];
                Top20FOM[npo]=Top20FOM[n];
            }
              Top20Switches[i]=SwitchID;
              Top20DC[i]=dc;
              Top20AC[i]=ac;
              Top20FPC[i]=tfpc;
              Top20FOM[i]=FOM;
              found=1;
           }
        }
        switch(SwitchID)
          {
             case 0x28882:
                   printf("\nNIST EC  ");
                     printf("SwitchID:%05lX\nDC: %4.3f AC: %d AC/TC:
%4.3f FPC: %d FPC/AC: %4.3f TC: %d FOM: %8.6f\n",

SwitchID,dc,ac,(float)ac/(float)TC,tfpc,(float)tfpc/(float)ac,TC,FOM);
               printf("Switches on: S1 S2 S4 S6 S9\n");
                     printf("Total Switches On: 5\n\n");
                     break;

             case 0x3CD92:
                   printf("NIST SSLF ");
                     printf("SwitchID:%05lX\nDC: %4.3f AC: %d AC/TC:
%4.3f FPC: %d FPC/AC: %4.3f TC: %d FOM: %8.6f\n",

SwitchID,dc,ac,(float)ac/(float)TC,tfpc,(float)tfpc/(float)ac,TC,FOM);
               printf("Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F7 S9\n");
                     printf("Total Switches On: 10\n\n");
                     break;

             case 0x3CDD3:
                   printf("SNAC      ");
                     printf("SwitchID:%05lX\nDC: %4.3f AC: %d AC/TC:
%4.3f FPC: %d FPC/AC: %4.3f TC: %d FOM: %8.6f\n",

SwitchID,dc,ac,(float)ac/(float)TC,tfpc,(float)tfpc/(float)ac,TC,FOM);
               printf("Switches on: S1 F1 S2 F2 S4 F4 F5 S6 F6 F7 S9
F9\n");
                     printf("Total Switches On: 12\n\n");
                     break;

             default:
                     break;
          }

     }

     printf("Top Twenty Switches based upon W1=%f W2=%f
W3=%f:\n\n",W1,W2,W3);
     for(i=0; i<20; i++)
     {
```

```
      printf("Switch Rank: %2d SwitchID:%05lX\nDC: %4.3f AC: %d AC/TC:
%4.3f FPC: %d FPC/AC: %4.3f TC: %d FOM: %8.6f\n",
          i+1,Top20Switches[i],Top20DC[i],Top20AC[i],
(float)Top20AC[i]/(float)TC,Top20FPC[i],(float)Top20FPC[i]/(float)Top20
AC[i],TC,Top20FOM[i]);

      totalsw=0;
      printf("Switches on:");
      for(j=0; j<18; j++)
        {
             if(SwitchIMask[j] & Top20Switches[i])
             {
                  totalsw++;
            switch(j)
                  {
                     case 0:
                           printf(" S1");
                           break;

                     case 1:
                           printf(" F1");
                           break;

                     case 2:
                           printf(" S2");
                           break;

                     case 3:
                           printf(" F2");
                           break;

                     case 4:
                           printf(" S3");
                           break;

                     case 5:
                           printf(" F3");
                           break;

                     case 6:
                           printf(" S4");
                           break;

                     case 7:
                           printf(" F4");
                           break;

                     case 8:
                           printf(" S5");
                           break;

                     case 9:
                           printf(" F5");
                           break;
```

```
                        case 10:
                              printf(" S6");
                              break;

                        case 11:
                              printf(" F6");
                              break;

                        case 12:
                              printf(" S7");
                              break;

                        case 13:
                              printf(" F7");
                              break;

                        case 14:
                              printf(" S8");
                              break;

                        case 15:
                              printf(" F8");
                              break;

                        case 16:
                              printf(" S9");
                              break;

                        case 17:
                              printf(" F9");
                              break;

                  default:
                              printf("Should Not Get Here");
                              exit(1);
                              break;

                     }
               }
         }
      printf("\nTotal Switches On: %d\n\n",totalsw);


   }
   return 0;
}
```

Bibliography

[1] "FIPS PUB 199, Federal Information Processing Standards Publication; Standards for Security Categorization of Federal Information and Information Systems," NIST, Feb 2004.

[2] "Federal Information Security Management Act," 2002.

[3] R. Shirley, "Internet Security Glossary, RFC 2828," 2000.

[4] "DOD Instruction 8500.2, Information Assurance (IA) Implementation," DoD, Ed., February 6, 2003.

[5] U. S. S. S. a. C. C. C. C. M. U. National Threat Assessment Center, "National Threat Assessment Center Report," November 2004.

[6] M. Keeney, E. Kowalski, D. Cappelli, A. Moore, T. Shimeall, and S. Rogers, "Insider Threat Study: Computer System Sabotage in Critical Infrastructure Sectors," United States Secret Service and CERT Program Software Engineering Institute, Carnegie Mellon University March 2005.

[7] L. A. Gordon, M. P. Loeb, W. Lucyshyn, and R. Richardson, "2005 CSI/FBI Computer Crime and Security Survey," Computer Security Institute, 2005.

[8] R. Richardson, "2003 CSI/FBI Computer Crime and Security Survey," Computer Security Institute, 2003.

[9] L. A. Gordon, M. P. Loeb, W. Lucyshyn, and R. Richardson, "2004 CSI/FBI Computer Crime and Security Survey," Computer Security Institute, 2004.

[10] R. H. Anderson, "Research and Development Initiatives Focused on Preventing, Detecting, and Responding to Insider Misuse of Critical Defense Information Systems," in *Workshop on Preventing, Detecting, and Responding to Malicious Insider Misuse*: National Security Research Division, RAND.

[11] "NSTISSAM INFOSEC /1-99, THE INSIDER THREAT TO U. S. GOVERNMENT INFORMATION SYSTEMS," NSTISSC, Ed., 1999.

[12] U. S. S. S. CSO Magazine, and Carnegie Mellon University Software Engineering Institute's CERT Coordination Center, "2004 E-Crime Watch Survey," 2004.

[13] S. Ramamoorti, "Internal Auditing: History, Evolution, and Prospects," in *The Institute of Internal Auditors*, 2003.

[14] R. Reagan, "Farewell Address to the Nation," January 11, 1989.

[15] "NIST Special Publication 800-12, An Introduction to Computer Security: The NIST Handbook," NIST, Ed.

[16] "COMPUTER SECURITY ACT OF 1987," in *Public Law 100-235 (H.R. 145)*, 1987.

[17] G. Stonenurner, A. Goguen, and A. Feringa, "Special Publication 800-30, Risk Management Guide for Information Technology Systems, Recommendations of the National Institute of Standards and Technology," NIST.

[18] R. C. Brackney and R. H. Anderson, "Understanding the Insider Threat, Proceedings of a March 2004 Workshop," 2004.

[19] "DoJ IG Hanssen Report, A Review of the FBI's Performance in Deterring, Detecting, and Investigating the Espionage Activities of Robert Philip Hanssen," Department of Justice August 2003.

[20] K. Dillard, J. Pfost, and S. Ryan, "The Security Risk Management Guide," Microsoft Corporation.

[21] S. Clark, M. Danseglio, K. Dillard, R. Harrison, J. Maldonado, B. Partridge, and T. Quinn, "Windows XP® Security Guide," vol. 2005, Version 2.1 ed: Microsoft Corporation.

[22] P. A. Porras, M. W. Fong, and A. Valdes, "A Mission-Impact-Based Approach to INFOSEC Alarm Correlation," SRI International.

[23] H. Debar and A. Wespi, "Aggregation and Correlation of Intrusion Detection Alerts," IBM Research, Zurich Research Laboratory.

[24] K. Mandia, C. Prosise, and M. Pepe, *Incident Response and Computer Forensics*: McGraw-Hill, 2003.

[25] R. Bickel, M. Cook, J. Haney, M. Kerr, U. CT01 T. Parker, and H. Parkes, "Guide to Securing Microsoft Windows XP®, Operational Network Evaluations Division of the Systems and Network Attack Center (*SNAC*)," *NSA*, Ed., Version: 1.1 ed.

[26] "How to view and manage event logs in Event Viewer in Windows XP®," in *Microsoft Technet.*

[27] M. Danseglio, K. Dillard, J. Maldonado, and P. Robichaux, "Threats and Countermeasures: Security Settings in Windows® Server 2003 and Windows XP®," vol. 2005, Version 2.0 ed: Microsoft Corporation.

[28] "Windows XP® Professional Resource Kit," in *Microsoft Technet*.

[29] A. Steven, "The Security Monitoring and Attack Detection Planning Guide," vol. 2005: Microsoft Corporation.

[30] M. Souppaya, K. Kent, and P. M. Johnson, "NIST Special Publication 800-68, Guidance for Securing Microsoft Windows XP® Systems for IT Professionals: A NIST Security Configuration Checklist, Recommendations of the National Institute of Standards and Technology," N. I. o. S. a. Technology, Ed.: U.S. Department of Commerce.

[31] M. W. Sullivan, "NATIONAL SECURITY AGENCY (*NSA*) SYSTEMS AND NETWORK ATTACK CENTER (*SNAC*) SECURITY GUIDES VERSUS KNOWN WORMS," in *Department of Electrical and Computer Engineering, Graduate School of Engineering and Management*: Air Force Institute of Technology.

[32] X. Yin, K. Lakkaraju, Y. Li, and W. Yurcik, "Selecting Log Data Sources to Correlate Attack Traces for Computer Network Security: Preliminary Results," University of Illinois.

[33] A. Valdes and K. Skinner, "An Approach to Sensor Correlation," DARPA.

[34] A. Valdes and K. Skinner, "Probablistic Alert Correlation," DARPA.

[35] P. Ning and D. Xu, "Adapting Query Optimization Techniques for Efficient Intrusion Alert Correlation," North Carolina State University.

[36] C. Kruegel, T. Toth, and C. Kerer, "Decentralized Event Correlation for Intrusion Detection," Distributed Systems Group, Technical University of Vienna 2002.

[37] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*: John Wiley and Sons, Inc., 1991.

[38] A. G. Anderson, *From Set Through Function: Elementary Mathematics for the Nonspecialist*: Wadsworth Publishing Company, 1972.

[39] K. Holsinger, "Decision Making Under Uncertainty: Statistical Decision Theory" University of Connecticut, 2005

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From – To)*
13-06-2006	Master's Thesis	Aug. 2005 – Jun. 2006

4. TITLE AND SUBTITLE

Development of a Methodology for Customizing Insider Threat Auditing On A Microsoft Windows XP® Operating System

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Levoy, Terry E., Gunnery Sergeant, USMC

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GIA/ENG/06-07

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; Distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Most organizations are aware that threats from trusted insiders pose a great risk to their organization and are very difficult to protect against. Auditing is recognized as an effective technique to detect malicious insider activities. However, current auditing methods are typically applied with a one-size-fits-all approach and may not be an appropriate mitigation strategy, especially towards insider threats.

This research develops a 4-step methodology for designing a customized auditing template for a Microsoft Windows XP® operating system. Two tailoring methods are presented which evaluate both by category and and by configuration. Also developed are various metrics and weighting factors as a mechanism to evaluate auditing effectiveness for the purpose of optimizing the template according to organizational security requirements. Various industry standard auditing templates are evaluated against a custom designed template. Results indicate that a customized auditing template tailored for an insider threat scenario is more effective at detecting insider malicious activities.

15. SUBJECT TERMS

insider threat, auditing, auditing template, security template

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Robert F. Mills(ENG) Robert.Mills@afit.edu
U	**U**	**U**	**UU**	81	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-6565, ext 4527

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18