

An Investigation into the Human/Computer Interface from a Security Perspective

by

Daniel J. Cross

A Thesis

Presented to the Faculty of

Bucknell University

In Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science with Honors in Computer Science and Engineering

3 May 2005

Approved By: _____

Luiz Felipe Perrone
Thesis Advisor

Gary Haggard
Chair, Department of Computer Science

Adaptability lies within the limits of understanding

Table of Contents

Acknowledgements.....	vi
Abstract	viii
Introduction.....	- 1 -
Part I User-induced Vulnerabilities.....	- 3 -
Chapter 1 – Email Attachments and Downloadable Programs: The clever candy we can’t resist.....	- 4 -
1.1 Various types of malicious software	- 4 -
1.2 Antivirus software and firewalls.....	- 11 -
1.3 Past threats	- 17 -
Chapter Summary	- 18 -
Chapter 2 – Passwords: Smart people with limited vocabularies.....	- 20 -
2.1 “Bad” Passwords.....	- 21 -
2.2 “Good” Passwords	- 26 -
2.3 Other potential issues.....	- 29 -
Chapter Summary	- 30 -
Chapter 3 – Social engineering: To be or... wait, what was the question?	- 32 -
3.1 How does it happen?	- 32 -
3.2 What are the different approaches?	- 33 -
3.3 Current Events.....	- 41 -
Chapter Summary	- 42 -

Chapter 4 – Users with malicious intent: Keep your enemies closer	- 44 -
4.1 The typical attacker	- 45 -
4.2 Increased Vulnerability.....	- 47 -
4.3 Examples.....	- 50 -
Chapter Summary	- 52 -
Part II Improved Methods	- 54 -
Chapter 5 – Ethical Attributes: Patterns of User Behavior	- 55 -
5.1 Utilitarianism	- 55 -
5.2 Kantian ethics.....	- 56 -
5.3 Utilitarian and Kantian ethics applied to computer security	- 57 -
Chapter Summary	- 59 -
Chapter 6 – Addressing User Behavior: Improving the mirror’s image.....	- 61 -
6.1 Potential solutions for the benign user	- 61 -
6.2 Potential solutions for malicious users	- 71 -
6.3 Case Study – Bucknell University.....	- 75 -
6.4 Suggested System Implementation and Technology Policy	- 80 -
Chapter Summary	- 94 -
Concluding Remarks.....	- 96 -
Bibliography	- 99 -

Acknowledgements

I would first like to take a moment to thank those who lent their expertise, love, and support while I undertook this endeavor.

No amount of thanks is enough to bestow upon my thesis advisor, Luiz Felipe Perrone. He represented my greatest resource, harshest critic, and staunchest ally throughout the entire process. From commas to concepts, he helped me pull all of this together and, without him, I question whether this thesis would have come to fruition. Thanks again to my advisor, mentor, and friend.

Thank you to my family and girlfriend who pledged their unrelenting love and support even before I had a topic and did not falter for a moment throughout. Shortened phone calls, researching and writing over breaks and still you all stayed behind me. Thank you.

My academic advisor, Xiannong Meng, has been a godsend throughout this thesis and my undergraduate career. The encouragement, dedication, and personal commitment he has provided me is nothing short of remarkable.

My professor, Patricia Wenner, has provided me with confidence and support since I began working with her. The ultimate end of a student-teacher relationship is when they can work side by side as equals. I first experienced this relationship with her and will not forget it. It is hard to believe we did not meet until halfway through my junior year.

Thank you, Ruth Miller. You are always willing to help in whatever way possible

and were a pleasure to stop and talk to. I can only hope that the department secretary for my graduate program is even half as kind and helpful as you have been over the past four years.

Thanks are in order for the head of the department, Gary Haggard. His willingness to take a more personal role in the development of this thesis was very much appreciated and his support, though understated, never undervalued.

Special thanks are in order for Eric Smith. Without him, there would be no case study of Bucknell University. His wealth of information expanded my knowledge beyond the scope of this thesis.

Thank you to the Department of Computer Science for instilling within me not just technical knowledge, but also the broader social considerations associated with this field and degree.

Lastly, I would like to thank my defense committee members Prof. Felipe Perrone, Prof. Matthew Higgins, and Prof. Rick Zaccone for reviewing my thesis and providing me with their invaluable wisdom and feedback.

Abstract

Despite the push for increasingly secure software and the popular opinion that secure software means secure systems, computer system users represent the greatest threat to a computer system's security. This text is not the first to consider the impact that users have upon the security of a system. However, prior attempts to elaborate upon this impact have focused on only one or two particular ways the user affects the system. This thesis establishes four different ways users signify a substantial risk to the security of a computer system: malicious software, poor passwords, social engineering, and malicious insiders.

In addition to identifying the potential problems posed by users, this thesis investigates how these threats can be mitigated. Utilitarian and Kantian ethics are defined and then applied to computer security. As is often the case, the best ethics, with regards to system security, that can be adopted by a user lie in a combination of the two modes of thought. A technology policy and system implementation that attempts to reduce the threats created by users is suggested at the end and represents the most significant part of the thesis.

Introduction

Every year, computer systems are compromised by any one of a variety of methods. Many experts and the public at large have put the onus of maintaining secure systems squarely on the shoulders of software manufacturers. While software companies are responsible for flaws in their applications, the heavy focus upon them shifts the responsibility for maintaining a secure system from those who pose the greatest threat: the users.

This thesis will examine the threat users pose to computers systems and methods and policies to limit that threat. Part I, titled “User-induced Vulnerabilities,” details various ways by which a user can compromise a system. The first chapter is titled “Email Attachments and Downloadable Programs.” It explains the concept of malicious software, the use of antivirus software and firewalls to combat it, and examples of previous malicious software threats.

The second chapter, referred to simply as “Passwords,” defines the advantages and disadvantages of both “good” and “bad” passwords. It also examines other ways that make the passwords selected insecure.

“Social Engineering,” the third chapter of the first part, focuses on the non-technical attack of an attacker. The chapter takes a look at how attackers employing social engineering take advantage of the user, the different approaches that can be used to do so, and an example of a recent, high-profile, social engineering attack.

Chapter 4, “Users with Malicious Intent,” places emphasis upon inside attackers

by investigating traits of the typical insider, the greater threat an insider represents, and examples illuminating the increased risk associated with insiders.

Part II is titled “Improved Methods” and describes how the risk users create can be mitigated. Chapter 5, titled “Ethical Attributes,” takes a peek at the two predominant ethical models associated with the work place, Utilitarianism and Kantian ethics, and applies the two modes of thought to computer security.

The final chapter, “Addressing User Behavior,” is the most substantial and important chapter in the thesis. This chapter suggests potential solutions to mitigate the threats posed by both benignly and maliciously intended users. Bucknell University’s system implementation and policy is used as a case study examining how a system can be secure and still allow for the free flow of information. Finally, the last chapter suggests a technology policy and implementation options to account for and improve upon user behavior and the threat it represents for computer systems.

Part I

User-induced Vulnerabilities

Chapter 1

Email Attachments and Downloadable Programs:

The clever candy we can't resist

We have all heard the phrase “Don't take candy from strangers,” as kids. As adults, it seems we just cannot resist. Obviously, we're not taking candy from people we don't know. However, we do open email attachments from people we do not know and, as a result, our systems get “sick” with viruses. Unfortunately, it doesn't stop there. Some of the free downloadable programs available on the Internet that are supposed to make life easier are exactly the reason why systems slow down to a crawl and pop-ups bombard computer screens every time a link is clicked. Why are we so careful with candy and so careless with computers?

The threat of malicious code has been increasingly publicized over the past decade. As a result, nearly all examples of malicious software are popularly referred to as viruses. However, viruses are a specific type under a more broad classification of malicious programs commonly referred to as malware.

1.1 Various types of malicious software

As briefly highlighted above, it is obvious that there are various types of malware that

can be used in an attack. Below, each type of malware is extensively defined in accordance with the classification previously described.

1.1.1 Worms

Worms infect other systems by self-replicating and take advantage of computers that are networked together. This, however, does not mean that the computer has to be connected through a router or switch to get infected. Any computer connected to the Internet is connected to a network and is thus vulnerable to such an attack. Worms do not infect the programs on the computer. They use address books and chat room addresses stored on the infected computer to infect other computers. Once on a computer, the worm will carry out the function it was prescribed to do. For instance, the Code Red worm was designed to conduct a denial of service attack after propagating [34]. Due to their frenetic propagation rate, worms often cause systems to slow down to a crawl. In short, worms are self-replicating, requiring no help from the user in their propagation.

1.1.2 Viruses

Viruses, unlike worms, are not autonomous in nature and depend on human interaction to propagate. Furthermore, a computer virus, like viruses that infect humans, cannot survive on its own. A computer virus must have a host. The virus, once in a host, infects other files and disks on a computer whenever the application it has infected is run

or the infected computer boots up. Viruses themselves are divided into 3 subgroups: boot-sector viruses, file viruses, and macro viruses.

Boot-sector viruses infect the boot-sector of a floppy disk or the partition table of a hard drive. When a computer boots up, the virus is loaded into memory. From here, the virus can infect other disks in the system. Removable media, such as floppy disks, can also be infected by boot-sector viruses. Currently, boot-sector viruses are less of a threat than they have been in the past. The substantial decline in the use of floppy disks, particularly for booting up, has declined significantly over the past decade or so. In addition boot incompatibilities with Windows 95 also impaired the spread of such viruses [4].

File viruses, on the other hand, infect various programs. An infected application, when run, puts the virus in memory. Depending on the file virus, this either results in the infection of other files within the same directory or the infection of the other programs in memory [7]. One would assume that, since programs are obviously very much a part of computing, file viruses would be thriving. However, file infectors have, for the most part, been extinct since the introduction of Windows 3.1, because they crashed the operating system and thus couldn't spread [4].

The final subgroup of viruses consists of *macro viruses*. Macros are small portions of code contained within documents. They are used to automate certain functions within these documents. Macro viruses take advantage of applications that allow the implementation of macros, such as Microsoft Word, Microsoft Excel, and Emacs, a UNIX-based text editor. Consequently, macro viruses are dependent upon the

language understood by the applications, not the operating system itself [43]. A single macro virus can infect and affect computers with different operating systems. The code for the virus is implemented as a macro [4]. Thus, when the document is opened, the embedded macro virus executes. Macro viruses, unlike the other types of viruses previously listed, have not experienced a drop-off over the years. To the contrary, macro viruses have become increasingly popular. This increase in popularity is due to the decline in boot-sector and file viruses and to the increased functionality and power of macros [35]. The sharing of documents has become increasingly popular thereby also increasing the ease with which macro viruses can spread.

1.1.3 Trojan Horses

Trojan horses, the final type of malware, are purposefully placed within a system. Trojan horse programs take their cue from the ancient story. For those not familiar, the Greeks, who had been at war with the Trojans, left behind a giant wooden horse as a peace offering as they made their way to their ships. The Trojans accepted the enormous gift and brought the horse, and the Greek soldiers hidden within it, inside of the walls of the city of Troy. The rest of the Greek army returned in the dead of night, entered the city after the soldiers from within the horse opened the gates, and sacked Troy. The malware referred to as Trojan horses, or simply Trojans, behave in a similar fashion. Trojans are placed within applications that appear to have a useful function. Once these infected applications are executed, the malicious code that makes up the Trojan also

begins executing thereby compromising the system. It is important to note that Trojans are initially implemented within the host program and distributed with it already embedded. They are not added subsequently, and are thus distinguished from file viruses [7]. The method of propagation also distinguishes Trojans from file viruses. While the active viruses propagate themselves throughout the system, activated Trojans do not generally infect other applications. Trojans are usually downloaded by the user. That is not to say that the programs that contain Trojans explicitly say: "WARNING: Trojan horses are contained in this program." However, the programs that contain Trojans make themselves enticing by advertising their uses. Users, willing to take advantage of the new features of this program, will download the program and install it, thus infecting their systems with the Trojan.

Software bombs, like Trojans, are also selectively installed on systems. The key difference between software bombs and Trojans is that, unlike the latter, software bombs are often installed by an attacker of the system. Software bombs are sub-categorized into logic bombs, time bombs, and letter bombs. Logic bombs are designed to go off, that is damage the system, once a particular event takes place. The event can be any one of a number of things, such as the deletion of a name from a payroll system [7]. Time bombs are designed to inflict damage on the system when a particular date and time is reached. A bomb could be set to go off at 7 P.M. on a Friday so as to incur the maximum amount of damage before it's discovered. Letter bombs differ from logic and time bombs by where they're hidden. Letter bombs are typically hidden in a file and are triggered when the file is read by a particular application [6]. While this may sound similar to a file

virus, keep in mind that the letter bomb generally must be intentionally placed on the system.

Web content and Internet browser plug-ins can also harbor malicious code. Such malicious content falls under the category of Trojan horse programs since the code propagates based upon its selection. However, unlike Trojans and software bombs, the malicious code does not exist within a standalone program. Computer systems utilizing web browsers are susceptible to weaknesses that can be taken advantage of by scripting languages. Scripting languages are high-level programming languages used to create increased functionality among software components [40]. Scripting languages are also used for value checking in documents filled out online and automating browser functionality, such as opening new windows and adjusting browser settings [4].

JavaScript is a scripting language developed by Netscape that can be embedded within Web pages [37]. It utilizes Document Object Models, a standard supported by the World Wide Web Consortium that represents “structured documents in a platform- and language-neutral manner [41].” In short, this means that JavaScript is used primarily for providing increased functionality and interactivity with online documents. Though useful for many web pages, past implementations of JavaScript have taken advantage of browser weaknesses and carries out such tasks as reading files off of the user’s system. While that particular problem and other subsequent vulnerabilities have been addressed and corrected, new implementations of JavaScript continue to find previously undiscovered holes [4].

On the other hand, Microsoft’s ActiveX code forces the user to decide if the

additional code should be executed [38]. Though this appears to be an improvement over JavaScript, the use of ActiveX is in fact a double-edged sword. On the one hand, the user is given the opportunity to ignore the additional code, thereby preserving the security of the system. However, on the other hand, ActiveX is more powerful than JavaScript and has the capability to do anything once it is executed [4]. However, Active X is not a scripting language in itself. Rather, Active X is implemented with scripting languages that facilitate the interoperability of software, thus employing what is known as the Component Object Model [42]. It is this interoperability that makes Active X software so powerful. Unfortunately, Active X interoperability is so powerful that, unchecked, it can wreak havoc upon a system. If every user of the system can be trusted to make the right decision every time, ActiveX works. If not, ActiveX controls present a significant security risk.

The precise method by which malicious programs are classified is largely disputed. Peter Neumann's Computer Related Risks [6] defines a Trojan horse as "an entity that contains code or something interpretable as code, which when executed can have undesired effects." Neumann then goes on to describe logic bombs, time bombs, letter bombs, and viruses as types of Trojan horses. Under this definition, worms used for malicious purposes also qualify as Trojan horses. Cybercrime, by Steven Furnell [7], separates malware into Trojan horses, viruses, worms, and software bombs, including time bombs and logic bombs. In Bruce Schneier's Secrets and Lies [4], malicious software is categorized by the means by which it propagates. This definition separates malware into

worms, viruses, and Trojan horses. Software bombs are classified as a subset of Trojan horses. The subtle difference between software bombs and Trojan horses is as follows: Software bombs are intentionally placed on a user's system by another party whereas the user intentionally places a Trojan horse on his or her own system. The remainder of this text will operate under Schneier's definition.

1.2 Antivirus software and firewalls

In this section, we define and discuss antivirus software and firewalls, which are the primary lines of defense against malicious software. Most networked systems employ firewalls and nearly all systems, public and private, employ antivirus protection. However, antivirus software and firewalls are far from perfect. In order to understand their weaknesses, how they operate must be clearly defined.

1.2.1 Antivirus software

Each worm, virus, and Trojan has a specific code that reveals its identity. Whereas in biological terms this might be a strand of RNA, in computers it is a specific sequence of instructions or bytes. This byte sequence represents the signature of the malware. One way antivirus software can safeguard a system is by scanning the executable files on the system and looking for these signatures. Antivirus software maintains a catalog of known virus signatures. If the software detects a signature it can identify, then it notifies the user that there is a virus on the system and then prompts the user what actions he or

she would like to take. These actions usually include an attempt to repair the file by removing the virus. If that's not possible, quarantining the file or removing it altogether are other actions taken by the software [7].

Another tactic employed by antivirus software is *immunization*. In the classic biological sense, immunizing a subject means inoculating the subject with a weakened sample of a virus so that a resistance is built up to it. Similarly, immunizing a computer system requires injecting part of the signature of the virus into the executable files present. Viruses are designed to not infect the same file more than once. By placing parts of the signatures of viruses in the executables, viruses can be fooled into thinking that the files have already been infected and will not try to infect those files. Thus, the system will be immunized [8].

A *checksummer* can also be used by antivirus programs to detect viruses. Data about each executable, such as size, is pushed through a one-way function, the results of which are maintained in what is called a checksum. Thus, checksummers do not need to maintain records of the various virus signatures. Instead, they maintain a record of the executables on the system and of the previously calculated checksums. When the antivirus software runs, it looks up the executables and again pushes its properties through the function. A computed checksum different than the checksum stored indicates a virus has been detected [8].

Unfortunately, these methods aren't bulletproof. The checksummer is perhaps the weakest method of the three. The checksummer is vulnerable to *stealth viruses* [8]. Stealth viruses operate by attempting to cover their tracks and hide. One way they can do

this is by recording the original size of the executable and then intercepting requests for the size. The virus simply has to return the original value instead of the current value in order to fool the checksummer [7].

Despite being more effective, scanning and immunizing also have a glaring weakness. In biology, viruses that do not exist cannot be immunized against. Furthermore, viruses that do not have a clear signature cannot be fully diagnosed and addressed. In a similar way, antivirus software implementing scanning and immunizing can only protect against viruses that are well-known. New viruses can attack systems freely until a signature is identified and widely disseminated. While antivirus software companies are improving the rate at which they make these new signatures available, a number of systems are inevitably infected beforehand. It is mathematically impossible to exhaust the possible number of computer virus signatures. As a result, this problem will never completely go away. Unfortunately, the biological comparisons do not stop with new viruses. Some viruses, such as the flu virus, mutate over time. Consequently, new vaccinations are required each year. A similar mutation capability can be built into a computer virus. Such a virus is referred to as a *polymorphic virus*. A polymorphic virus mutates as it propagates in an attempt to avoid leaving a signature. The rate of mutation is obviously much faster in computer viruses than their biological counterparts. This mutation can take place by either a rearrangement of code or by a different encryption of the code with each propagation. If the code is encrypted, a small part of the virus code is then used to decrypt it [7].

Antivirus software alone cannot adequately protect a system. As a result, it is often coupled with firewalls to present a more effective line of defense.

1.2.2 Firewalls

Firewalls stand as barriers between a computer system and a network. A firewall is designed to block harmful programs and code from getting to a system. As such, a firewall is generally the first line of defense. Consider the security checks at airports that prevent unauthorized people from passing through to the gates and waiting airplanes. Similarly, firewalls prevent unauthorized information from penetrating a system. A firewall can be implemented at three different levels. These levels include packet filtering, circuit gateways, and application relays.

Packet filtering involves allowing or denying information to pass through based on its packet address and port number. Every information packet has an address and port number. Similarly, every person approaching the security check has, or should have, a form of photo identification and a ticket. Someone whose picture on the identification does not match the person's face or whose plane is not waiting at the terminal beyond the check (such as if someone were trying to access terminal B when his or her plane leaves gate C3) would be denied. Information packets with invalid addresses or those attempting to access the system via a port it should not be going through will be blocked [8]. When information is passed over a network, it is not transferred as a single large chunk of data. Instead, the data is broken into small chunks, or packets, of data. These information packets have an address associated with their origin and, depending on that

address, can only pass through the firewall through a specific door, or port. Thus, an address trying to go through a port it should not go through will be blocked.

Firewalls that utilize *circuit gateways*, though similar, are more comprehensive than those that utilize packet filtering. Circuit gateways recompile the information from the individual information packets. Thus, instead of letting each packet pass through the firewall as it arrives, the firewall blocks all of the packets until the last one arrives, at which point the firewall checks the address and port number of the data. Consider again the airport analogy and assume that a traveler wished to pass through the security check, but his carry-on bags were coming after him separately. Instead of allowing the traveler through and then simply passing through the carry-ons when they arrived as a firewall implementing packet filtering would do, the traveler would be forced to wait at the security check until his bags arrived. Then, the traveler and his carry-ons would be checked as a whole. The firewall then checks the packet address and port number of the entire packet and determines the validity of the fragments collectively [8].

Application relays are more thorough than even circuit gateways. True to their name, application relays are able to block information packets intended for a particular application. They can deny the active content of a Web page and macros in documents. Consider the airport security check scenario again. In this case, the traveler cannot simply pass through with his luggage even after supplying the proper identification and trying to access the right terminal. The traveler and his baggage must be searched for anything that could harm the travelers or planes beyond. Anything that is found during this search is removed from the traveler before he or she can pass through. An

information packet is similarly stripped of its blocked content if an application relay is implemented by the firewall [8].

However, as with antivirus software, none of the three firewall options are perfect. Take for instance packet filtering: Packets are fragmented in such a way that, once the first packet gets through, following packets can modify the properties of the first packet. In doing this, a packet with an invalid address can penetrate the systems firewall [8]. This text previously stated that firewalls using circuit gateways were more comprehensive than those using packet filters. Indeed, the circuit gateways's practice of piecing the information packet together before checking it does prevent the attack mentioned above. Unfortunately though, it does not prevent malicious code in the information packet from reaching the system if a valid address and port number were supplied. Then why not application relays? As listed above, they do block active content and macros. Despite this advantage, firewalls implemented with application relays face another enemy: time. Look at the security check example once more. Currently, American airports implement (or at least they should) the equivalent of application relays at their security checks. Travelers must present proper identification, try to access the proper terminal, have their carry-on baggage with them, and subject themselves to searching by hand, x-rays, metal detectors, or some combination thereof. As a result of these elaborate security measures, when the airport sees a lot of traffic, a line develops at the security check. This hinders the time efficiency of the travelers to some degree. Implementing a firewall with application relays can create a similar bottleneck that hinders the efficiency of the users.

1.3 Past threats

Malicious software is far from a recent concept. Over the years it has consistently and persistently evolved as measures are taken to safeguard systems against it. It is oft said that the future is a reflection of the past. For this reason, we will revisit a few past examples of malware before anticipating the future of malware. These past examples will include the *Melissa virus* and the *MyDoom worm*.

The Melissa virus was a macro virus that infected computer systems in March of 1999. The virus used macros to automate the propagation of the virus. The macros of one Microsoft Office application can be used in another. The Melissa virus took advantage of this by embedding code in an attachment that, once opened, would automatically send email with the infected file as the attachment. A generic title and message were added to the emails sent. This virus blurred the line between worms and viruses by using malicious macros to automate its propagation. The federal government, when pressing charges against the author of the virus, claimed the limit of \$80 million in damages [11]. However, the 2000 Computer Virus Prevalence Survey put out by ICSALabs [12] places estimated damages around \$385 million.

The original MyDoom worm struck in January of 2004 and, since that date, more than 30 variants have been created. The worm propagated via email and through networks. The worm initiated a denial of service attack against the SCO Group's Web server. However, the MyDoom worm also implemented a backdoor in the system that enabled other computer users to remotely access the system through ports 3127 – 3198.

Thus, we see a worm coupled with a backdoor typically seen in Trojan horses. Though it is obvious that the primary target of the worm was the SCO group, the worm compromised every system it infected by opening the backdoor [9]. Furthermore, it was estimated that the MyDoom worm “caused \$1.52 billion in U.S. damages and \$4 billion in worldwide losses – based on cost of lost revenue, labor, hardware and software [10].”

The above two examples both indicate a growing trend that was addressed here at the beginning: the lines between different types of malicious software are growing increasingly blurred. Authors of malicious software are taking advantage of the favorable attributes of various types of malware to create what seem to be malware chimeras. Combined with the continual adaptations authors of malware make, malicious software only promises to become more potent and widespread in the future.

Chapter Summary

The chapter began by defining malicious software and categorizing it. After several different categorization examples, this text defined the categorization to be dependent on the method of propagation of the malicious software and that these categories are worms, viruses, and Trojan horses.

The chapter’s first section described each category of malicious software in detail. Worms propagate entirely on their own whereas viruses require some unknowing interaction by the user. Trojan horses, on the other hand, are placed on a system when a user knowingly installs and executes a program with a Trojan horse hidden inside.

Software bombs are types of Trojan horses that are knowingly installed by someone other than the common user. Malicious code written as JavaScript and ActiveX controls also fall under the realm of Trojan horses.

Section two discussed antivirus software and firewalls. The section is divided into two subsections, one for antivirus software and one for firewalls. Each subsection discusses the different ways in which the relevant piece of software can be operated. The weaknesses of both antivirus software and firewalls are also discussed in their respective subsections. Antivirus software can only protect systems from viruses that are already known and likely attacked other systems. Firewalls are confronted with a performance versus time conundrum and can be thwarted by Trojan horses that create tunnels in them.

The final section gave examples of malicious software that became widespread. The MyDoom worm and Melissa virus inflicted millions of dollars in damages. More disturbingly however, they blurred the line between different categories of malware indicating that authors are adapting their code to combine the advantages of each.

Chapter 2

Passwords:

Smart people with limited vocabularies

People once thought passwords were only used by secret societies and government agencies. However, this notion became antiquated with the advent of computers. Many currently possess passwords for their email accounts, website registrations, online bank accounts, employee accounts, etc, though the term “passwords” is used loosely. Often, people will use the same password for all, or at least some, of their accounts. Failing this, people will use variations of the same password to “mix it up.” There is no formal password training link when logging on to a website. Truth be told, most people do not know how to pick a secure password. Ross Schneier points out, “As bad as passwords are, users will go out of the way to make it worse. If you ask them to choose a password, they’ll choose a lousy one... One study of actual passwords found that 16 percent of them were three characters or less, and 86 percent of them were easily crackable [4].” A perfect example of this can be found in Kevin Mitnick’s book The Art of Deception [1]. While discussing a social engineering attack on a state’s Department of Motor Vehicles, Mitnick describes how an attacker was attempting to gain access to the phone switch that handled calls going into the DMV, but discovered that the switch was password protected. How did he get in? The switch’s password was associated with updating the

software so, on his fourth try, the attacker tried “update” and succeeded. This game of guess and check can be applied on a larger scale too. An attacker can employ similar methods using the file containing all encrypted passwords for the system. By using the same encryption function the system uses to encrypt, the encrypted form of different passwords guessed can be compared to the encrypted passwords in the file. A match indicates that the password attempted was valid. Insecure passwords place any system people log into at risk.

2.1 “Bad” Passwords

Statistically speaking, most passwords selected by the user are bad passwords. In this case, a “bad” password is defined as a password that can be cracked rather easily. Bad is in quotes because there are advantages to even a bad password, but that will come a little later in this text. Generally speaking, the more restricted the character set, the easier it will be to decipher, or *crack*, the password. Thus, passwords that rely strictly on the English alphabet are easy to crack. An alphanumeric character set makes it a little more difficult to crack the password, but not significantly. Try varying the case of the letters? We’re getting there, but it’s still too easy. Toss in the other characters available on a keyboard and, though still crackable, the password’s security is improving. Now stop and consider how many passwords you use that change the case of the letters and include numbers and other characters, such as those used for punctuation. A common response to this would be: “I use my dog’s name and I capitalize the first letter. Does that count?”

No. Unfortunately, most selected passwords are common. They are based on words or on names; sometimes a birth date is tacked onto the beginning or the end. These types of passwords are highly vulnerable to what is referred to as a *dictionary attack*. The dictionary in a dictionary attack is not a digital version of Webster's Dictionary. It does not necessarily contain all of the words in the English language. In fact, the dictionary most often will include more than just words. The dictionary often contains both common names and common pet names in addition to common words. There are some dictionaries that include modified strings of words, including intentional misspelling of words. These modified strings include the entries in the dictionary written backwards or with numbers representing letters, i.e. 4 represents "A," 0 represents "O," or 5 represents "S." Other dictionaries can be added to include more English words or foreign words. Due to poor password selection, many passwords can be cracked in a matter of hours, days, or weeks. While a time in the scale of hours will concern most, many are not worried about a password that takes weeks to crack. Who would spend weeks trying to break into a system? Since the computer does all of the work, it's not particularly difficult for someone with even a little motivation. Combine this with the fact that the damage that could be inflicted once the password is cracked could take months or even years to recover from and, suddenly, weeks do not seem like a long period of time.

Bad passwords are also often short. Many passwords are required to be only 6 characters in length. Though it is easier to remember a password drawn from a larger character set when it is short, this also makes it vulnerable to another form of attack. A brute force attack is the most straightforward attack. The person trying to crack a

password using this method will try every possible combination of characters until the password is cracked. For example, assuming that the password was drawn from a character set that included just upper and lower case letters, the passwords “A” – “z” would be tried, followed by “AA” – “zz,” “AAA” – “zzz,” and so on until the password was cracked. The larger the character set, the longer it will take to crack the password. However, a password of six characters or fewer will be cracked nearly all of the time, regardless of the character set. In fact, one study conducted by the Cambridge University Computer Laboratory [3] determined that passwords consisting of six or fewer characters could be cracked in a reasonably short time using a brute force attack.

Passwords are also vulnerable to *birthday attacks* [30]. Contrary to its name, a birthday attack has nothing to do with the birthdays of its victims. The name comes from a mathematical phenomenon known as *the birthday problem* [29]. The birthday problem asks the question, “Assuming that a person is equally likely to be born on any given day of the calendar year, how big does a group of people need to be for there to be at least a 50% chance that two or more people share the same birthday?” Many assume the answer to this question is 183, just over half the number of days in a year (excluding leap year). Obviously, if all 183 were different, then the chances that the 184th person had a birthday held by another person would be $\frac{183}{365}$, or 50.1%. However, the group only has to have 23 people. This is understandably hard to believe and deserves another moment of evaluation. Since leap year is excluded, there are 365 possible birthdays, which will be represented by the letter *b*. Assuming there is already one person in the group, the chance

that the second person has a different birthday is equal to $\frac{b-1}{b}$. Similarly, the chance that the third person has a different birthday is $\frac{b-2}{b}$, the fourth person's chance is $\frac{b-3}{b}$, and so on. Thus, the n^{th} person's chance of having a different birthday is $\frac{b-n}{b}$. These probabilities combine when considering the group as a whole. Therefore, the chance that everyone in a group of people has a different birthday can be represented by:

$$\Pr\{\text{no repeated birthdays}\} = \frac{(b-1)(b-2)\dots(b-n)}{b^{n-1}},$$

which can be simplified to

$$\Pr\{\text{no repeated birthdays}\} = \frac{b!}{(b-n)!b^n}.$$

Remember this is the probability that no one in the group has the same birthday. The probability that at least two people in the group do have the same birthday is given by

$$\Pr\{\text{at least two identical birthdays}\} = 1 - \Pr\{\text{no repeated birthdays}\},$$

or

$$\Pr\{\text{at least two identical birthdays}\} = 1 - \frac{b!}{(b-n)!b^n}.$$

Given that the result of this expression needs to be greater than $\frac{1}{2}$ and that b is equal to 365, we can solve for the smallest n using the inequality

$$\frac{1}{2} < 1 - \frac{365!}{(365-n)!365^n}.$$

The resulting computation yields $n = 23$. The result from this inequality can also be

approximated by the expression $1.2\sqrt{b}$. From a password security standpoint, this means that an attacker randomly guessing passwords can obtain a valid password after only $1.2\sqrt{b}$ tries. For instance, assuming that there are a million million possible passwords, an attacker would only have to guess 1.2 million times to have a greater than 50% chance of one of his guesses being an active password. While that may seem like a lot, considering the speeds at which computers operate, 1.2 million is not substantial. Some authentication system implementations incorporate a delay, thus decreasing the rate at which password guesses can be made. However, not all authentication systems employ a delay.

Despite their shortfalls, bad passwords have one crucial advantage: they're easy to remember. The very qualities that make bad passwords bad also give them their distinct advantage. Common words and names can be easily associated with friends, family, events, nicknames, etc. Numbers tacked onto the end of words can represent dates such as birthdays and anniversaries. Using a password that is somehow linked personally to the user, though easier to guess, is also much easier to remember. The same Cambridge [3] study referred to previously also revealed that among users given little direction and allowed to select their own password, it took an average of 0.7 weeks for the users to remember their password. These passwords are thus so easy to remember that users often do not need to record them. This fact decreases the chances the password will be discovered by non-technical means.

2.2 “Good” Passwords

It has long been believed by various members within the security community that randomly-generated passwords are the best. However, when discussing randomly-generated passwords, it is important to first introduce the concept of *entropy*. Bruce Schneier states, “...in the context of cryptography, it [entropy] is a measure of uncertainty. The more uncertain something is the more entropy in that thing [4].” The entropy of a password can be measured through the use of combinatorics. Combinatorics can be used to determine how many different combinations exist for a set number of distinct variables. For instance, there are four, 2-character combinations using the letters A and B: AA, AB, BA, and BB. Similarly, there are eight, 3-character combinations of A and B: AAA, AAB, ABA, BAA, ABB, BAB, BBA, and BBB. The possible number of combinations can be determined mathematically by raising the number of distinct characters, 2 (A and B), to a power equal to the number of characters in each combination, 2 and 3 in the examples above, respectively. So, a 2-character combination using the letters A and B should have $2^2 = 4$ possible combinations and a 3-character combination using the letters A and B should have $2^3 = 8$ possible combinations. The greater the number of possible combinations, the greater the entropy of the password is. A randomly-generated password has the highest possible entropy. High entropy is advantageous for passwords, because it increases the average amount of time it would take to crack the password.

Entropy can be maximized in any one of three ways: lengthening the password, selecting a truly random group of characters, and expanding the character set.

Lengthening the password: The entropy of a password can be increased by making it longer. That is, an eight character password has greater entropy than a six character password. Ideally, a password is made as long as the system will allow.

Selecting a truly random group of characters: Another way to increase the entropy is a system is to select a truly random group of characters. In order to do this, each character must be selected from the full set of possible characters. For example, assume the first character selected is “c.” The next character selected could also be a “c.” No characters are removed from the character set after being selected. This allows for the highest possible entropy within the set. In a character set that includes only upper and lower case alphabetic characters there are 53,459,728,531,456 possible eight character combinations (26 upper case letters + 26 lower case letters = 52 distinct characters $52^8 = 53,459,728,531,456$). There are 256 different ways to capitalize an eight-character word. Since many attackers using dictionary attacks often account for the word to be written backwards, there are 512 different ways to represent an eight-letter word. Oxford contends that there are approximately a quarter of a million distinct English words [5]. If every word were eight characters long, the dictionary for a dictionary attack would

contain approximately 128 million entries. This is only .00024% of the possible eight-character combinations.

Expand the character set: The other way to increase entropy is to expand the character set. For instance, if numbers are included in the character set for the previous example, the chances of a random set of eight characters being in the “attack” dictionary drops one order of magnitude from .00024% to .000059%. Expanding the set even further to include all of the possible characters on a keyboard, the chances plummet to .0000021%. Thus, a randomly generated password is far more likely to be resilient to a dictionary attack. Furthermore, the sheer number of eight-character combinations prevents, or at least discourages, brute force attacks.

However, similar to the previous section, this section has “good” in quotes. In this case, this is done because, while advantageous in many ways, randomly-generated passwords have a distinct disadvantage: they are very difficult to remember. Results from the aforementioned study at Cambridge University [3] revealed that, among students assigned a randomly-generated, eight-character password, an average of 4.8 weeks passed before a student had to stop carrying a paper record of the password with them. In other words, it took, on average, longer than a month for a student to consistently remember his or her password. However, this also points out a related problem: the student had to carry the password around for almost 5 weeks. While the paper record makes retrieving the password easier for the student, it also makes it easier

for the attacker. Copying, stealing, or simply picking up a misplaced or lost password compromises the account and thus the system. Say for instance a new employee writes down his or her password to log onto a system and tapes it to the bottom of a keyboard. An attacker, especially an insider, could go to the desk while unoccupied and obtain the password. Now, any attack launched from that account points to the new employee instead of the insider who obtained the new employee's password.

2.3 Other potential issues

Other potential issues are associated with password selection, several of which were alluded to in this chapter's introduction. Many passwords are reused over and over again by the same person on different systems. Therefore, cracking the password on one system exposes several systems, thereby creating a domino effect. In this way, it is possible for an attacker to gain access to your system without initially implementing a dictionary, brute force, or birthday attack.

Another problem includes how often users change their password. Some systems require users to change their passwords periodically. However, one study illuminated a couple of the creative ways users adopted when forced to change their password. In some instances, users tack the numerical value of the current month to the end of the password. For example, John Smith's password might be "Smith09" in September and "Smith10" in October. Another clever method involved changing passwords until the original password worked again. John Smith, in this instance, would change his password twice. The first password would be meaningless. He would then immediately

change his password back to the original password. Some systems keep a backlog of passwords selected by a user. However, the lengths of these logs are not infinite. John Smith could simply continue this process of changing to a dummy password then back to the original until the original was no longer in the logs and thus would be accepted [5].

How passwords are changed presents still another problem. In some cases, a phone call to the system administrator is all that is needed to change a password. An instance of this occurred in which an attacker, wanting more information, specifically the source code, about the encryption software a company developed, needed access to the system. He called up the computer center and managed to obtain the random-access code he would need to access the system. He did so by only dropping a few names. While this instance technically falls under social engineering, which will be discussed in the next chapter, it nonetheless highlights the importance of proper authentication when giving passwords to employees [1].

Chapter Summary

Passwords, designed and implemented to restrict access to sensitive data, are often one of the greatest weaknesses in a system. This glaring weakness is due more to password selection than to poor encryption methods and password storage. Most people do not select secure passwords, thus leaving the system they log in to vulnerable to attack.

The passwords most users select for themselves are “bad.” Bad passwords are often based on a limited character set, relatively short, and based off of a word or name

the user likes or knows. As a result, they are vulnerable to often birthday, brute force and dictionary attacks. Birthday attacks are particularly successful against low-entropy passwords. Brute force attacks are particularly useful against passwords that are short in length. One example showed that all passwords six characters in length could be cracked in a reasonable amount of time using this method. The dictionary attack stores common words, names, and pet names, as well as different ways to capitalize them and backwards orientations of them. Though a small subset of the total names and words available, these attacks are often successful against password protected systems. Though that they can be remembered easily is an advantage, the commonality of the words used for passwords also gives them their vulnerability.

Some passwords, especially those that are assigned to users are “good.” Randomly-generated passwords are much harder to crack than the passwords users select for themselves. Passwords improve as the level of entropy increases. A password’s entropy can be increased by making the password longer, ensuring a completely random selection of characters, and enlarging the character set. Unfortunately, the inherent randomness also makes these passwords difficult to remember. As a result, many users are required to record the password and store it in an insecure location, such as a desk drawer or coat pocket, in order to be able to access the system. This makes the system vulnerable.

Other problems with password selection include their reuse across multiple systems, how users change their password, and how passwords are changed.

Chapter 3

Social engineering:

To be or... wait, what was the question?

There is no such thing as a stupid question, but there is such a thing as a dangerous question. Many computer security policies assume that the attack will be made through a computer. In fact, the threats discussed thus far in this thesis also adhere to this theme. However, some of the most insidious attacks occur without the attacker using a computer at all: *Social engineering* is a method by which an attacker can obtain critical information to compromise a system without an electronic attack. Such attacks are commonly successful because not only do they catch their target unaware and off guard, they often leave their target just as unaware after the attack.

3.1 How does it happen?

In a social engineering attack, the attacker convinces someone within the organization to supply the desired information. The social engineers, those who carry out social engineering attacks, are often polite, friendly, and patient. In fact, they often leave you wishing you were contacted by people like them more often. Social engineers are a pleasure to talk to. They often brighten your day. And they obtain even your most vital

information by simply asking for it.

The attack plan of a social engineer involves, above all else, gaining trust and illustrates the fact that the weakest link in security is people. In this case, the social engineer takes advantage of the willingness of people to trust others. A smile, a friendly voice, and patience make people want to believe the social engineer. Some general knowledge about the organization and some carefully dropped key words are often all that's needed to acquire the information.

If gaining trust is the most important part of the attack plan for a social engineer, the need for help is a close second. The propensity of people to help others plays into the hands of a skillful social engineer. Any difficulty in gaining trust can often be glossed over by giving the impression that help is needed. Many people feel sympathetic to others who need help and are more willing to bend the rules as a result. Given that some positions within organizations require helpful, outgoing people, safeguarding to make sure employees are not too helpful can be particularly difficult.

3.2 What are the different approaches?

The art of social engineering has various forms. *Dumpster diving* is the most impersonal method, because it involves sorting through discarded material in attempt to attain information. However, it also arouses the least amount of suspicion and can open the door for future attacks. The worst-case scenario is that highly sensitive and confidential information has been discarded. However, many companies already shred their classified

documents. Where then is the threat? Any discarded company phone lists or phonebooks allow the social engineer to choose his point of contact. Documents that give some indication of company hierarchy not only give the attacker an idea of whom to target, but also a bit of authenticity. For instance, say an attacker got a hold of the hierarchy for the department and maybe a phone number or two. The social engineer may call and say, “Hi, Dorothy? This is Greg in Accounts Receivable. I was hired to work for Mr. Roberts just a few days ago. Anyway, I’ve got Mr. Roberts on another line asking for an account number, but I’ve lost the info in this stack of new hire paperwork. You know how it is. Could you help me out?” It’s not much, but simply knowing that there is a Mr. Roberts in Accounts Receivable is often enough to gain the trust of a “fellow” employee. Another potential risk involves discarded CDs, hard drives, back-up disks, etc. These items cannot be subjected to shredding and thus may contain potentially vital information still intact. A dumpster diver’s greatest advantage comes not from the relative seclusion under which dumpster diving must take place. It virtually guarantees that the attacker cannot be identified by either his voice or physical characteristics. The plethora of seemingly innocuous information and the near certainty of anonymity combine to establish a credible threat to the system and organization as a whole.

Another way to obtain sensitive information is through direct contact. More to the point, the attacker physically goes to the company to access the system. Most attackers avoid this route, because they can be physically identified if the attack is linked back to them. Furthermore, such conclusive identification essentially burns the source. That is, a social engineer that makes a direct contact attack on a system often cannot

make any subsequent direct system attacks on the same system. Direct contact attacks on a series of similar systems (say a group of banks) can result in notification of other banks, further compromising the attacker. That said, a social engineer who is not concerned with burning the source and has not commonly been identified before may attempt this method. For example, a social engineer walks into a company office. He claims to be from the help-desk as he posts fliers on bulletin boards around the office. The fliers all have the new phone number for the help-desk on them. The attacker leaves after running out of fliers and the employees all assume that he simply went back to the help-desk. Over the following days, as the users encounter various problems, they call this new phone number on the fliers. Thinking they're going to reach the help-desk, the employees instead reach the attacker who, in reality, posted his own phone number. The attacker is then able to obtain sensitive information, such as passwords, from the users without arousing any suspicion [4]. This attack arouses virtually no suspicion, despite the direct contact, and significantly jeopardizes the security of the system. Furthermore, the backdoor installation nearly ensures that the social engineer does not need to return to the organization.

Another instance of direct contact involves a couple of teenagers who wanted to reach the production floor of a helicopter manufacturer. After calling headquarters for the organization in Phoenix pretending to be working for a marketing firm, the kids had the name of an employee, his supervisor, and both of their extension numbers, which were conveniently left on voice mail recordings. With this information, they called the production facility, made a late-night appointment for their arrival, and toured the

production floor [1]. These teenagers only wanted to look, but what could a more experienced attacker with different aims have achieved? The direct attack, carefully executed, will enable the social engineer to obtain, or have the means to obtain, the vital information he or she desires.

The most common method of attack in social engineering is the phone call. A person calling for seemingly innocuous information may be setting you up for an attack. This method requires the attacker to interact with one or more employees of the target organization. However, while it requires vocal contact, a physical presence is not required and thus a degree of anonymity is maintained. Attacker can attempt to coax the information out of their target in any number of ways. One popular method to extract information from a target is to insist that there is a problem or crisis and that the solution of it depends on the person at the other end of the phone. Creating a situation with a problem adds both an air of importance to the situation and gives the impression that the social engineer is under stress. This creates two situations that favor the attacker. The importance of the situation imposed upon the person on the other end places pressure squarely on his or her shoulders. He or she often does not want to be responsible for exacerbating the situation. Furthermore, the urgency of the situation combined with the stress the recipient of the call (henceforth referred to as the “mark”) is now experiencing may cause them to bypass normal protocol. Normal protocol may include asking for an identifying code, confirming with a supervisor, or even simply refusing. Creating exceptional circumstances may cause the mark to think that exceptions will need to be made. The stress the social engineer feigns experiencing plays to the human willingness,

and thus also the mark's, to help those that are in trouble. The combination of the mark's desire to help the attacker and to not exacerbate the problem results in the mark releasing the requested information. For instance, Kevin Mitnick [1] describes a situation in which the attacker dials one of a phone company's private numbers and says,

Hey, this is Paul Anthony. I'm a cable splicer. Listen, a terminal box out here got fried in a fire. Cops think some creep tried to burn his own house down for the insurance. They got me out here alone trying to rewire this entire two hundred-pair terminal. I could really use the some help right now. What facilities should be working at 6723 South Main?

The woman who answered felt bad for the attacker and, thinking it was ok to make an exception every once in a while, agreed to tell the social engineer the information. Not all social engineering attempts over the phone have to put stress on people though. For instance, an attacker called a bank to verify the term for the identifier the bank uses with a credit union. The attacker pretended to be an author, said he was researching for a book, and the person on the other line was more than happy to help. The attacker then called another person in the same bank claiming to be from the credit union and was conducting a survey. Amidst the seemingly benign questions included a question about which 800-number the bank uses to call the credit union and which merchant ID had been assigned to them [1]. These two simple questions enable the attacker to look up the account of nearly anyone for whom he has the information. Given the ease with which

he obtained the information from the bank, obtaining personal information about people is hardly a reach.

Another method to extract information is via what's called the *reverse sting*. Under this method, the social engineer gives the impression of authority. More to the point, the social engineer assumes authority over the mark. In doing so, the mark feels more comfortable releasing the information. Surely another person with as much or more authority is allowed to have the same information. Unfortunately, as with the other methods, no organization is immune to this type of attack. Kevin Mitnick describes a scenario in which an attacker phones a bank and obtains its branch number. The attacker then calls another branch of the same bank and determines both the name of a teller and the time at which she will leave on her lunch break. Calling back during the teller's lunch break, the attacker speaks to another bank employee. This time, he insists that he is from the other branch that that he is supposed to fax an urgent message to the teller who's on break. He manages to convince the employee not just that it needs to be faxed, but also that the employee needs to give him the authentication codes since he's the one sending the fax. Armed with the authentication codes, the social engineer then called another branch and had money transferred to an account in his name [1].

Another, more complicated instance involved an attacker who managed to discover one of the phone numbers local law enforcement used for calling the Department of Motor Vehicles, getting access to the phone switch into the DMV from its own staff, and forwarding phone calls using a little technical know-how to a disposable cell phone. He got the phone number for the DMV calling the sheriff's office,

transferring to Teletype, and stating that he had been using a number to call DMV and it wasn't working. Only the last 4 digits of the number were wrong so the recipient volunteered the phone number. The attacker called the DMV pretending to be from Nortel, said that they were constructing a database to do remote upgrades, and thus obtained the number to access the switch. The attacker then simply forwarded the calls to his disposable phone. Ready to go, the attacker was able to obtain personal information of police officers when they called. After obtaining the information he was looking for, he pretended that his computer crashed so they would not expect the information they were looking for. He then used the personal information of the police officer to obtain license numbers and other data from the DMV [1]. The effectiveness of these reverse sting examples on banks and law enforcement, two groups that we would hope are unequivocally secure, should serve as a reminder of the importance of proper authentication.

Phishing is a social engineering attack that attempts to bypass authentication methods by requesting sensitive information through email. The social engineer employing phishing techniques will send an email purporting to be an employee of a bank or credit card company, a system administrator, or in some other position requiring knowledge of sensitive information. The attacker, consistent with other forms of social engineering, often impresses upon the mark that some problem has occurred and that his or her bank account number, password, social security number, employee ID, etc is necessary to remedy the situation. An example of such an email follows:

From: bob.helpdesk@yahoo.com
To: kevin.foreman@myCompany.com
Subject: Your employee account

Kevin,

I regret to inform you that your password was compromised last night. As a result, we need you to temporarily change your password to "edcrfvtgb." We will inform you when you can again select your password.

Thank you for your cooperation,

Robert

IT Department

The attacker presented a problem and attempted to convince the employee to change his or her password. The employee, willing to help out and scared his or her password had been compromised, would change the password, thereby compromising the system. Thus, it is imperative to remind users to be critical of the email they receive. Users need to verify the authenticity of the individual requesting the information.

3.3 Current Events

There are some people and organizations that insist that this will never happen to them. However, a current example of organizations' vulnerabilities to social engineering attacks has been made publicly known. ChoicePoint®, a company that considers itself to be a "leading provider of decision-making information that helps reduce fraud and mitigate risk [36]," has just recently announced that its system and the information stored on it has been compromised. Its database contains 19 billion public records. A press release on the company's website announces "This incident was not a breach of ChoicePoint's network or a "hacking" incident and did not involve any of ChoicePoint's customer information [2]." How then did this come about? A group of people established approximately 50 accounts with ChoicePoint in such a way so as to appear as legitimate businesses. In doing so, the social engineers were able to obtain the names, phone numbers, social security numbers, and credit reports of hundreds of consumers. In fact, ChoicePoint went so far as to send out warnings to over 145,000 people nationwide, including 35,000 in California. Unfortunately, only California currently requires businesses to notify consumers when their personal information has been compromised. Thus, it is entirely plausible to unknowingly be a potential victim of identity theft. Your co-workers, friends, or self may have been made vulnerable by a similar attack. Such information would be more than enough for an attacker to instill as sense of trust. How could someone disrupt the system with your identity?

Chapter Summary

The chapter started off by introducing the concept of social engineering and what role it has within the scope of this text.

It then moved on to learn that social engineering hinges upon two key factors: the development of trust and the willingness of people to help each other. The social engineer will turn these well-meaning traits into weakness in an attempt to obtain information from the system. The need for helpful, courteous employees in some positions makes safeguarding against social engineers increasingly difficult. To make things worse, the social engineer is often so polite and courteous that the organization remains clueless that their system has been compromised.

There are several different methods used to carry out social engineering attacks. Dumpster diving, the least personal of these methods, involves obtaining information, confidential or seemingly innocuous, from disposed papers, hard drives, and other media. On the other hand, a direct contact is the most personal. It involves the attacker physically entering the premises. Phone calls are the most common means by which attackers carry out social engineering attacks. Information can be easily obtained through this attack with only limited contact. The reverse sting requires the social engineer to convince the mark that he is in a position of authority or, at the very least, is cleared to know the information requested.

An example of a social engineering attack has recently made headlines. ChoicePoint® revealed that the personal information of hundreds of people had been

compromised by a group of individuals who set up fictitious, seemingly legitimate businesses to gain access to the information stored on their databases.

Chapter 4

Users with malicious intent:

Keep your enemies closer

We have discussed various ways in which users of a system can unwittingly compromise that system by selecting poor passwords, opening email attachments, downloading programs that include malicious code, and volunteering critical information as a result of social engineering. However, another significant threat remains: current and former users with malicious intent. Malicious insiders have a critical advantage over their external counterparts: they often have legitimate access to the system they are attacking. A malicious insider who attacks the system he or she works with is equivalent to a family member who robs his or her own house. There are no signs of forced entry, because the attacker has been trusted with the key. Malicious attacks made by insiders often inflict more damage since insiders are more aware of the inner workings of the organization. Similarly, the family member is familiar with the house, where valuables are stored, and what valuables are likely to be there. Bruce Schneier points out, “Most computer security measures... try to deal with the external attacker but are pretty much powerless against insiders [4].”

4.1 The typical attacker

The movie *Office Space* [22] presents the perfect example of the stereotypical inside attacker. The three key characters in the movie all seem to be in their late 20's or early 30's. The insiders are all males and they are all single. None of the attackers are financially well-off. However, that is not to say they are poor either. They simply do not think they are paid enough and want more money. Underappreciated and dissatisfied, the three co-workers decide to attack their own system by diverting the fractions of a cent left over from transactions into a different account. The three characters are all technically competent and combine to write and install the malicious code into the system. Thus, we have our stereotypical attackers: single males who are relatively young, technically competent, and dissatisfied, both financially and professionally. If the stereotype were completely accurate, discovering malicious insiders would be much easier than is typically the case.

The United States Secret Service and Carnegie Mellon's CERT® Coordination Center conducted a joint study [23] investigating computer-related insider attacks in the Banking and Finance Sector. The initial findings presented in the study were compiled from an analysis of insider attacks in the banking and finance sector between 1996 and 2002. In all, 23 incidents, and their combined 26 attackers, were researched. The study revealed that the ages of the attackers varied from 18 to 59. The lower half of this range fits stereotype. However, generally speaking, many people do not expect an employee in his or her fifties to be conducting attacks against a computer system. It seems that while

somewhat accurate, it is unfair and inaccurate to assume that malicious insiders are overwhelmingly young. The study also revealed that, while 52% of the attackers were single, 31% were married. Assuming the other 17% of the attackers were divorced, widowed, or in some other way resultantly single, there are still a significant number of malicious insiders who are married. Thus, while a popular assumption, the concept that malicious insiders are single is in fact a little misguided. Though many malicious insiders are single, the statistics are not so overwhelming as to support the stereotype. Although not all inside attackers are young or single, surely the large majority are male, right? In what is probably one its most surprising statistics, the CERT®/Secret Service study determined that females accounted for 42% of the perpetrators. Although the scope of the study is admittedly small, this figure is higher than many would think and certainly does not adhere to the stereotype of the malicious attacker. So far it seems that while a young, single male is likely to be the attacker in an inside incident, it is just as probable the attacker is a 55 year-old married woman. Of course, in any case involving computer-related crimes, the attacker is going to be technically competent. It turns out, however, that this most basic assumption does not even hold true. “In 87% of the cases studied, the insiders employed simple, legitimate user commands to carry out the incident,” and, “Only 23% of the insiders were employed in technical positions, with 17% of the insiders possessing system administrator/root access within the organization.” The study thus makes obvious that the technical users of a system are not the only users that can use their malicious intent to inflict damage on their organization. Still, it is important to remember that in that same study, the perpetrators who held technical positions were the

only users to sabotage the system. In fact, it seems the only attribute of the stereotype that fully holds up under the scrutiny of this study is the financial situation of the perpetrators. Though 81% of insiders attacked at least partially due to the prospect of financial rewards, only 27% of the attackers “were experiencing financial difficulty at the time of the incident.” Other motives, such as revenge and dissatisfaction, also factored into some instances, however not as prominently or consistently as the potential for monetary benefit. Thus, the stereotype does hold up when considering the motives of the attacker. Other key statistics were accurately represented within the movie *Office Space*, though they are not explicitly aspects of the stereotype. Most attackers (81%) planned their attack in advance. Furthermore, 85% of malicious insiders informed at least one other individual of the attack. Despite this however, only “twenty-seven percent of insiders had come to the attention of a supervisor and/or coworker for some concerning behavior prior to the incident. Thus, profiling inside attackers is difficult and often inaccurate.

4.2 Increased Vulnerability

As alluded to at the beginning of this chapter, many computer systems face increased vulnerability from insider attacks. The system vulnerabilities discussed thus far in this thesis can be exploited to an even greater extent by insiders.

Email attachments and downloadable programs represent a significant threat and are particularly useful to insider attackers. Though some firewalls can be configured to

prevent certain information, file types, and programs from leaving a system, the majority of them are configured for keeping those same potential threats out. Furthermore, firewalls are often setup to protect the system as a whole, not the individual computers of the system from each other. The antivirus programs employed on most systems are far from perfect as well. An inside attacker is likely to have knowledge of when the antivirus software scans the system. Those inside attackers that are highly technically educated can write a virus or worm that possesses a signature which antivirus software cannot yet identify. Furthermore, an inside attacker with knowledge of when the antivirus software programs are executed could release such a virus or worm at a time that would cause it to remain undetected for the longest time and cause the most damage. Such weaknesses in firewalls and antivirus software make systems vulnerable to inside attacks. Malicious users can also implant software bombs and backdoors with greater ease. Rather than produce a program with a trapdoor hidden within it and hope that it gets downloaded, the inside attacker can implant the malicious code directly into the system, though with varying degrees of difficulty.

Password authentication is, in some cases, the easiest vulnerability of which an insider can take advantage. It is obvious that in many cases a user of the system can access the system by utilizing his or her password. However, other password weaknesses can be exploited. Poor password selection or generation can enable an inside attacker to conduct his or her attack under the guise of another user. Passwords selected by users are vulnerable, particularly if the users are not private about their password selection method(s). As stated previously in the passwords chapter, many users select passwords

that are personal in nature. The casual and personal relationships that develop between employees make it easier for insiders to determine the passwords of other users. However, passwords assigned by the system administrator, IT professional, etc. can be even more vulnerable to insider attacks. If policies requiring users to replace the default password assigned to them with a personally selected password are not enforced, an insider can take advantage and assume the identity of any user account that still employs a default password. However, default passwords are not the only assigned passwords that can be exploited. Randomly-generated passwords are also vulnerable to insider attack. Recall the difficulty involved with remembering a string of random characters. Many users write down such passwords as a result. In instances such as this, an insider might only need to look at a note next to the keyboard of or inside a desk drawer near a deserted workstation to obtain a password. All of a system's passwords become vulnerable if an insider is able to access the password file. The password file, perhaps obviously, contains the passwords for users of the system, but it also contains user IDs, home directories, and username information. The password file has to be world-readable, because applications on the machine refer to the information in it during their execution [28]. When users enter their passwords to login to the system, the system encrypts the password and checks it against the encrypted passwords in the password file. If the password entered is in the password file and the password entered is associated with the username, then the user is granted access to the system. A malicious insider who gains access to the password file can launch dictionary, birthday, and brute force attacks against the file to determine the passwords of the users in the system. The ease with

which some passwords can be obtained also enables attackers to gain greater user privileges in some cases, thereby placing the system at an even greater risk.

Social engineering also presents a more substantial risk when employed by an insider with malicious intent. Where social engineering attacks from the outside hinge upon gaining the trust of the person they contact, insiders employing social engineering tactics often already have the trust of the person they contact. Though instances exist that still require the insider to gain some trust from a fellow coworker, insiders, more often than not, find gaining this trust easier than their external counterparts do. The insider attacker often knows the precise terminology that would instill trust. In addition, many employers have some record or access to the company hierarchy. As in any social engineering attack, any little bit of information to indicate that “we’re all on the same team” can be enough to instill the trust necessary to obtain the desired information. Unfortunately, malicious users almost always have many such little bits of information.

4.3 Examples

Many security analysts believe that there are more insider attacks made on companies than are reported in the public media. Why are not all insider attacks made public? In some cases, the amount of money lost due to the attack is less than the amount of money the company could lose as a result of public fallout. Thus, companies attempt to deal with the problem quietly and out of sight from the public eye. Nevertheless, examples of insider activity abound.

Nearly a decade ago in July of 1996, a logic bomb developed by Timothy Lloyd crippled his former employer, Omega Engineering Systems. Timothy Lloyd had been employed by Omega Engineering Systems for eleven years and was chief computer network program designer prior to the attack. However, poor reviews ultimately led to his dismissal. The impending firing early in the July of 1996 left a bitter taste with Lloyd. As a result, prior to his dismissal, Lloyd employed a software time bomb that went off three weeks after his termination date. The bomb deleted approximately 1000 programs the company used for manufacturing. Lloyd also removed the server's back-up hard drives and local installations of the programs from workstations. All of this activity went unnoticed, because Lloyd was "the only Omega employee responsible for maintaining, securing and backing up the file server [13]." Consequently, Omega Engineering Systems sustained more than \$10 million in damages. The company was also forced to layoff 80 workers and lost its footing in the market as a direct result of the attack [13].

Another disgruntled employee, this one Roger Duronio, attacked his former employer, UBS PaineWebber in March of 2002. Duronio, a 60 year-old computer systems administrator, grew increasingly upset over his salary and bonuses. This growing resentment ultimately resulted in his resignation. However, he did not go peacefully. Prior to leaving the company, Duronio's ability to access the UBS PaineWebber's entire network allowed him to place a time bomb that affected approximately 1000 of the company's computers, two-thirds of its networked workstations. The bomb "detonated" at the prescribed time on March 4, 2002 and

removed files from the more than 1000 computers. The expense of the damages inflicted exceeded \$3 million [14].

Unlike the previous two examples, Jessica Sabathia was not a disgruntled employee. Even more interesting, Sabathia was not employed in a technical position nor did she have any substantial technical background. Indicted in May of 2004, Sabathia was an accounts payable clerk for the not-for-profit organization North Bay Health Care Group. She caused \$875,035 in damages by writing 127 different checks to herself. Attorneys for the Department of Justice reported that “to conceal the fraud, SABATHIA altered the electronic check register to make it appear that the checks had been payable to North Bay's vendors [15].” Thus, while the more damaging attacks were perpetrated by technically knowledgeable attackers, insiders without such information still pose a formidable electronic threat.

Chapter Summary

Most preventative security measures focus on preventing an external attacker from attacking a system. Unfortunately, this same focus leaves many organizations susceptible to insider attacks. Attacks made by malicious users are often more damaging than attacks made by users outside of the system due to the greater familiarity the insiders have with the system and the ease with which they can access the system.

The first section defined, or more accurately, debunked the stereotype of the inside attacker. While the single, young, technically competent male looking for more

money is a legitimate suspect, data collected by the United States Secret Service and Carnegie Mellon's CERT coordination center initially indicates that this stereotype is not accurate. Perpetrators in the banking and finance center were of varying ages, and socioeconomic and educational backgrounds. Attackers were also fairly evenly balanced with regards to their sex.

The increased vulnerability of systems was explored in depth in the next section. Risks that accompany email attachments and downloaded programs are magnified by malicious insiders. Most firewalls are designed to keep malicious software out, not prevent it from moving around within the system. As for antivirus software, an insider with knowledge of when the software scans the system can release a virus at a time that will inflict maximum damage. Malicious users also already have a password to the system they are trying to exploit thereby making their entry into the system seamless. Furthermore, insiders are often aware of default passwords distributed to employees and can more easily access work areas where passwords may have been written down. They also often find it easier to gain the trust of employees thereby making their social engineering attacks harder to notice.

Reports of insider attacks can be difficult to come by given that public backlash can further hurt the organization already attacked. Examples still remain however. The attackers in the examples used varied in age, sex, and technical proficiency. Damages from the three examples combined to exceed \$13 million.

Part II

Improved Methods

Chapter 5

Ethical Attributes:

Patterns of User Behavior

Ethics have a pervasive influence within the workplace. Furthermore, the ethics of system users can have a significant impact on a systems security. Do your users favor utilitarianism or Kantian ethics? Which does your organization try to foster? Which is better for running a tight ship and which one lets things slip? The prospect of trying to influence your users' ethics is by all means a controversial one. However, simply knowing their moralistic tendencies can help you account for their actions and reactions.

5.1 Utilitarianism

Utilitarianism hinges on what can essentially be called a cost-benefit analysis. Utilitarianism, a form of consequentialism, relies on the concept of acting based on what is right for all involved. No single person or group gets priority over any other. The cost or benefit is calculated for each person independently. The course of action chosen will often support whichever option benefits the most people [25]. Thus, this branch of ethics is almost democratic in nature. In many ways, this can be good for an organization. However, utilitarianism is not without its flaws.

In order to create a cost-benefit analysis, a value must be associated with everything. While perhaps well-meaning, the assignment of value creates problems. The assignment unfairly depends upon the individual to correctly, objectively gauge the potential cost or benefit for a host of people whose cost and benefit may differ subjectively. Furthermore, the concept of value and use of cost-benefit analysis inexorably leads to an assignment of price. This leads to the possibility that unethical actions can be bought. In effect, an “end justify the means” code of conduct pervades. The concept of justice is skewed by this mantra. Rather than trying to effect equal benefit and cost for everyone, utilitarianism requires only overall the consideration of what is most beneficial, regardless of the level of cost required of certain individuals [26]. Furthermore, utilitarian ethics does not distinguish between good actions and bad actions, only good results and bad results. Thus, the utilitarian mode of thought supports achieving what is best regardless of the morality of the actions required [25].

5.2 Kantian ethics

Kantian ethics, contrary to utilitarianism, does not rely upon the determination of consequence. Instead, Kantian ethics focuses on the morality of the action itself. Emotion is viewed as weak, because it can be distorted to justify selfish acts. As a result, reason determines the course of action [26]. This reason relies upon objective thought of the morality of the actions involved. Thus, an individual operating under Kantian ethics, given the choice between speaking the truth and lying, will pick truth every time.

Nonetheless, this absolute focus upon the morality of the action presents problems.

Kantian ethics ignores the outcome of actions. It assumes that the most favorable result will be produced as a result of universality. Universality is defined as the concept of acting in a way that you would have everyone else act [26]. The concept of universality is encompassed by the phrase “Do unto others as you would have them do unto you.” Following this, if everyone acts based upon what is right, no wrong can come about. Unfortunately, this idea is utopian in nature and falls apart when hostile entities are introduced. Furthermore, the course of action becomes unclear when there is no definitive right or wrong answer. Take for example an instance of conjoined twins where collectively they cannot survive, but separating them kills one or the other. How do you choose which child will live and which will not? Kantian ethics do not delineate how to proceed with such a decision. Furthermore, not all people possess the same morality. Kant does not allow for the idea that an individual’s perception of morality is shaped by his or her upbringing. As a result, an act perceived to be moral and good by one individual may be perceived to be immoral by another individual [25].

5.3 Utilitarian and Kantian ethics applied to computer security

The consideration of consequence is important when looking to keep information secure. The negative impact that results from releasing secure information discourages information leaks. Users who adhere to utilitarianism and are asked to supply information will consider the effect their actions have upon their organization, co-

workers, attacker, the public, and themselves. Recognizing the threats from malware, poor passwords, social engineering, and inside attackers, users will often act in a way that maintains the systems security. However, this cannot be guaranteed. Consider again the assignment of value to actions. In cases where a user considers taking on an insider role, a user may legitimize his or her actions by saying the benefit, including those to themselves, outweighed the costs of releasing such information. Utilitarianism also falls apart in the face of some social engineering attacks. Users may release information they think is insignificant based on the low value they assign to it. Similarly, users may use open email attachments, download programs, and select poor passwords, because they simply misestimate the value in doing so.

Kantian ethics would thus seem to be the better choice. A user with Kantian ethics will inform his superiors of an insider every time. In fact, that same user would likely notify his superiors of any actions that compromise the system. Unfortunately, the willingness of the user to act morally, without regard to the consequences, can also create security issues. The uneducated user may not have any objection to releasing information he or she does not understand to be important. Also, an attacker employing social engineering could present a moral need for secure information. For instance, the attacker could call a doctor's office or hospital demanding information on a patient and insisting that the life of the patient rests in his or her hands. Such a situation would present a moral conundrum for the recipient of the phone call, because on one hand they are not allowed to release the information, but on the other hand a death could result from withholding it.

Neither of these two ethical extremes provides the perfect answer. Ideally, users should possess a sense of ethics that takes into account both the morality of their actions and the results of their consequences. However, there is likely no perfect blend. A user whose ethics are more utilitarian may be better for one particular position than another. This holds true for users who favor Kantian ethics as well. Regardless, the ethics of users must be taken into account when examining the strengths and weaknesses of an organization. Doing so will enable the organization to anticipate how users will respond to certain situations and enable it to implement policy that anticipates these reactions.

Chapter Summary

This chapter addressed the ethics prevalent among system users. The focus is upon utilitarianism and Kantian ethics.

Utilitarianism weighs the costs and benefits of the end result. The course of action chosen is based upon the best result available. No regard is given to the means by which the goal is attained. Thus, utilitarianism can be defined by the mantra “The ends justify the means.” Furthermore, the distribution of sacrifice and reward is not considered.

Kantian ethics focus solely upon the action as opposed to the end result. A user adhering to Kantian ethics will always do the “right” thing. However, each action is treated like it is in a box. The consequences of the decision are not taken into account. As a result, the action chosen does not always yield the best results. In addition, the

existence of two, contradicting, moral decisions can yield unpredictable results.

There is not a blend of the two modes of ethics described here that would be perfect for every position. In fact, a user's ethics may make him or her better suited for one position than another. The assessment of users' ethics can be best applied to implement policy that can anticipate how users will act.

Chapter 6

Addressing User Behavior:

Improving the mirror's image

We have thus far explored the various ways in which user can compromise a system. Many problem solving solutions involve removing the factor that is causing the problem. In this case, that would require “userless,” or autonomous, machines. However, preventing users from interacting with systems also prevents work from getting done. It is self-evident that this is not a viable solution. Since users cannot be prevented from interacting with the system, what can be done? Is it enough to simply educate users? Do we simply account for poor user behavior? Can policies help protect systems from their users? As happens in most cases, the best solution lies within some combination of the three.

6.1 Potential solutions for the benign user

Users create system vulnerabilities due to practices that make them susceptible to malware, password, and social engineering attacks. However, the extent to which users unknowingly create risk for systems can be limited. Each of these vulnerabilities can

through good policy selection, user education, and system configuration.

6.1.1 Addressing Malware

This thesis's Email Attachments and Downloadable Programs chapter has already shed light on two types of software programs, firewalls and antivirus, that can help curb malicious software's impact on a system. Despite their shortfalls, firewalls and antivirus software still provide a crucial line of defense. Antivirus software, generally, can only protect against past viruses. However, promptly downloading updated virus definitions can significantly decrease the amount of time a system is vulnerable to a new virus or worm. For many antivirus programs, the process of downloading updated definitions can often be automated. The amount of time that passes between each check can often be adjusted. Furthermore, most antivirus programs also allow scans to be conducted autonomously. Thus, an antivirus program could be configured to check for updates at 3 AM and to scan the system at 3:30 AM. Such a setup would minimize worker interruptions and rely upon the computers clock instead of the memory of the system administrator.

An antivirus software program based upon checksumming could also be implemented. However, instead an antivirus program utilizing the relatively simple checksum function, an antivirus program using a cryptographic hash, such as MD5 or SHA1, could be utilized. These cryptographic hashes are one-way functions like their checksum counterparts. However, a cryptographic hash creates a new function result, or

hash, from the previous hash and the bytes that make up the next block of code. When the first block of code is pushed through the function, there is no previous result to push through it, so a predetermined hash is pushed through instead. However, this only happens the first time. Because each hash is dependent upon the previous hash, the function is collision resistant; that is to say it is much more difficult to push through different values and obtain the same result. Thus, an antivirus program employing a cryptographic hash is much more resilient than one using a checksum function [35]. This antivirus program could also be set to check the hashes of the executables at a designated time, thus freeing itself from running only when the system administrator remembers.

Firewalls, on the other hand, often impact user activity to some degree. For this reason, firewalls often have to be tailored to a particular system. An environment in which users pass a large amount of traffic passes through a firewall might be impaired by protection measures that take a significant period of time. In this instance, packet filtering might be employed. However, if users do not initiate a significant amount of the traffic passing the firewall, then an application relay maybe implemented given that the longer time processing time would have minimal impact on the users. Given the weaknesses of packet filtering and the potential loss of time, money, and public stature should the system be successfully attacked, no less than a firewall implementing circuit gateways should be utilized. This being said, an application relay implementation, providing it is considered economically feasible, is favored over a circuit gateway implementation. The ability to block active web content and macros is a valuable trait when protecting a system from attack.

Intrusion detection systems can also be utilized to limit the impacts of malicious software attacks. Also referred to as activity monitors, intrusion detection programs monitor the system on which they are employed for suspicious activity. Suspicious activity includes, among other things, the sending of many emails in a short time [16]. Intrusion detection mechanism can alert system administrators and IT staff to a potential problem on a machine. In some cases, this enables them to isolate the infected machine and prevent its spread within the system.

Intrusion detection systems can be divided into two main groups: misuse and anomaly. Misuse detection systems operate using a set of parameters representing typical user behavior. For instance, a misuse detection system can be used to keep track of the commands typically run on a particular machine. The use or attempted use of a command, application, or file not used by the user may indicate the computer is being controlled through a trapdoor. Anomaly detection systems do not operate based on predefined parameters, however. Anomaly detection systems look for unusual behavior. Commonly implemented with artificial intelligence, anomaly detection system “learn” what appropriate use is [8]. An anomaly detection system might notice that the password file is almost always accessed between 9 AM and 5 PM. Any malware probing for the file in the middle of the night would be inconsistent with the normal behavior of the system. As a result, the anomaly detection system would issue a warning. Intrusion detection systems can also be used to detect insider attacks.

There exists a category of programs for adware and spyware detection and removal. Adware posts advertisements, often in the form of pop-ups, on a computer.

Spyware tracks the habits of the user and relays that information back to an attacker. Similar in operation to antivirus software, the programs that detect adware and spyware focus their effort on identifying Trojan horses, tracking components, data-mining programs, and other forms of malicious software [17][18]. The parallels between adware and spyware detection programs and antivirus software continue, however. Most adware and spyware removal programs utilize a set of definitions or identifying characteristics to find and weed out adware and spyware from systems. Unfortunately, this fact means these programs also share the antivirus software's main weakness: they can only identify and remove adware and spyware whose characteristics have been determined and included in the definitions file. Updated definitions can often be obtained periodically. Some adware and spyware removal programs support scheduled updates as well as schedule scanning. Furthermore, many of these anti-adware and anti-spyware programs are available on the Internet, free of charge.

Another method to limit the threat of malicious software involves restricting user permissions. The permissions of users can be adjusted to restrict them from installing programs. While this may be unacceptable in some cases, denying installation privileges to users prevents them from installing software with malicious code embedded within it. Users in need of a new piece of software would then be forced to contact a system administrator who could safely obtain a program that performs the desired functions. This could put a significant amount of strain on some IT departments. However, instances where user privileges can be limited should be taken advantage of whenever possible.

6.1.2 Passwords

The passwords utilized by users represent another possible weakness in a system's security. While it is initially difficult to see what measures can be taken to better secure passwords, a variety of measures do in fact exist.

The policy of requiring passwords to be changed periodically is effective and should be maintained. Updating passwords, say once a month, gives a password a limited shelf life. Thus, even if an employee's account is compromised, the system is vulnerable only for a month, not the entire time that user is employed by the company. However, the company needs to aggressively discourage users from cycling through a variety of passwords once a month so that they can use their favorite again. To remedy this, the number of password changes per time period could be limited to around three. In any case, the number of allowed changes should be less than the number of most recent passwords stored for a particular user. Unfortunately, limiting the number of false entries also means a poor password could be the final allowable password for a given time period, thereby placing the network at increased risk. In this case, a user would often keep the insecure password rather than contact the IT department. To account for this, the IT department could launch dictionary attacks against company passwords to discover the weak passwords. The department could then require the noncompliant users to again change their password by making an exception with regards to the time period.

The company must also discourage users from adopting a similar password every

month. For example, a password that is blue01 in January, blue02 in February, blue03 in March, and so on leaves the system vulnerable. To solve this, new passwords could be checked against the passwords recently used. If the proposed new password has a substring of a set number of characters that is found in a previously used password, the proposed password would be rejected.

More policies are necessary however. Randomly-generated passwords are considered to be the most secure, whereas self-selected passwords are often the easiest to remember. Unfortunately, remember that self-selected passwords are often the easiest to crack and that randomly-generated passwords are the hardest to remember and are thus often written down. A remedy to this problem involves the use of mnemonic passwords. A mnemonic is defined by Webster's Dictionary [20] as an aid to help remember something. "Every good boy does fine" is a mnemonic to remember the five musical notes that exist on the lines in sheet music. A password can be developed using the first letters of a phrase. The phrase should be personal in nature and not popularly known. For instance, the phrase "I moved to Connecticut when I was 5" becomes "ImtCwIw5." The collection of first letters gives the appearance that the password is random. However, its personal nature is easy to remember. Other characters can be easily included as well, such as using '&' instead of 'a' for "and." Scott Granneman [27] suggests using a combination of upper case letters, lower case letters, numbers, and symbols. He insists a good password employs three of the four. The Cambridge University study [3] mentioned previously investigated the effectiveness of mnemonic-based, or "passphrase" passwords. The results of the study showed that fewer passphrase

passwords were cracked than randomly-generated passwords, though only by two percentage points. Thus it seems that mnemonic passwords are as secure as randomly-generated passwords of equal length. It is important to note that even passphrase passwords of six characters or less were cracked by brute force attack in the study. Therefore, a minimum character length of eight characters should be enforced [3]. The study also concluded that the users employing passphrase passwords need only .6 weeks to memorize their password in comparison with .7 weeks for those without guidance and 4.8 weeks for those whose passwords were randomly-generated [3]. Thus, mnemonic-based passwords prove to be easy to remember as well. Having the traits of being both easy to remember and the appearance of random generation, the mnemonic-based password is the recommended method by which passwords should be selected.

Another method exists for adding randomness to a password. This method is attractive, because it requires system implementation, not user compliance. Salt is often added to food to give it a little something extra. Salting a password essentially means the same thing. Instead of improving the taste, the salt adds to the randomness of the password. A salt is typically a short string associated with a user. This string is concatenated with the password prior to encryption thereby adding a random component to the encrypted password. Each user has a particular salt, though they do not know it. Thus, a pattern cannot be established across a few passwords to determine the salt. Salting a password requires no user intervention and is particularly useful even if the encrypted passwords strings become available to an attacker [21].

Another common mechanism used by modern systems can significantly limit the

effectiveness of automated attacks, such as brute force, dictionary and birthday. The system can be implemented in such a way that the verification process is slowed. The increasing speed of computers and their network connections enable thousands of passwords to be attempted every second. Computer users, while sensitive to the time it takes for their machines to perform a job, do not need an almost instantaneous response. Users are unable to distinguish between an event that occurs in $1/100,000^{\text{th}}$ of a second and an event that occurs in $1/10,000^{\text{th}}$ of a second. The password verification process can be slowed to a reasonable speed without significantly impacting the user. Forcing the verification process to take one second would force the number of passwords tried per second to one. Meanwhile, one second is a significantly short enough period of time to minimally impact the user. In fact, if the user used his or her password ten times per day, the user would wait a negligible 50 seconds over the course of an entire week.

6.1.3 Social Engineering

The non-technical nature of social engineering attacks forces most practical solutions to them to be non-technical in nature. Kevin Mitnick takes this a step further stating, “The truth is that there is no technology in the world that can prevent a social engineering attack [1].” That said, the greatest defense against the skilled social engineer is the educated employee. Education of the employees must occur in two parts. First, every new employee should be presented with information regarding social engineering tactics and practices. It is not enough to simply hand the new employee a pamphlet. Each new

employee should be presented this material in an interactive setting with another, already established employee. The second part of the education process is continuing the education with current employees. Many professionals are required to continue their education by completing a set number of hours every year. This requirement is often filled by attending classes at conferences, attending speaker presentation, and other events that will further their education. The goal is to keep professionals up-to-date with the latest knowledge in their field thereby making them better. Similarly, continuing to educate employees even after their hiring forces them to be aware of new tactics and prevents them from forgetting about old ones. Occasionally, but persistently, reminding employees of their responsibilities and of their importance to the company will encourage them to remain vigilant and instill a sense of trust. Educating and reminding the employees about the threats of social engineering attacks needs to be a persistent and prevalent part of the organization's culture.

A defined protocol for the release of information needs to be a part of this education process. The release of any information should only happen after the verification of the identity of the individual and his or her authorization to access the information. Identity verification can be carried out via an employee identification number. This number should be verified on an employee database. The supplier of the information can then ask a question pertaining to a published piece of personal information. There should be several pieces to choose from. Pieces of information should not be exceedingly personal, such as address, phone number, birth date, mother's maiden name, and other pieces of information that can be used for identity theft. Pieces

of information that can be used include college major, years with the company, and name of the user's first pet. It is important that this information is not used by the user in other verification systems, such as those used for home alarms and credit cards. A daily code can be used to verify that the individual is authorized to be given the information. As was the case in a previous example, the user supplying the information should first ask for one of several codes. Employees must remain firm if the requesting individual insists on supplying a different code. The requester must supply the precise code requested. The codes utilized must not operate on a cyclic basis. Each code should be generated randomly. Furthermore, any documents containing the daily codes should be promptly shredded at the end of the day. Whenever possible, the daily codes should be spread via a challenge-response system. In a challenge response system, a user types a specific code assigned to them into a device, the challenge. The device then returns a different code, the response [19]. This response can be used as the daily code. Furthermore, these devices can be set to provide a new code over an array of time periods. Thus, they can be set to change every half hour thereby making the system even more secure. These safeguards not only prevent outside attackers, but will also hinder the efforts of malicious insiders.

6.2 Potential solutions for malicious users

Malicious users present the unique problem that they are, in most cases, already familiar with the inner workings of the organization with which they are associated. As noted

before, many of the problems users pose are exacerbated when the user possesses malicious intent. Some of the potential solutions for benign users mentioned above will be helpful against malicious users. A malicious user can be prevented from intentionally installing damaging software by restricting the permissions of users. Intrusion detection programs can help isolate malicious software that originates on a malicious user's machine. The password solutions proposed would effectively prevent malicious users from gaining access to other user accounts. The use of mnemonic-based passwords means that passwords will not be written down nearly as often. Salting passwords can prevent the insider from determining the passwords of other user even if they possess the password file. The insider does not know his or her own salt string and even knowing would not help in determining the salt string of his or her co-workers. With regards to social engineering, an active, informative program that presses the need for adhering to the guidelines already set forth can significantly limit the successfulness of such attacks. Furthermore, the aforementioned study released by United States Secret Service and the CERT® Coordination Center [23] revealed that 65% of the attackers did not consider the harmful ramifications of their actions. Thus, the education program implemented would remind potential attackers of the consequences associated with attacking the system.

However, despite these solutions, further improvements are necessary to protect organizations from their most potent threat. The most common mistake on the behalf of companies is their failure to promptly remove the access privileges of a former user. It seems obvious that terminated or soon to be terminated employees could turn malicious in the days near their firing, yet too often organizations leave their accounts active.

Revoking the accounts of former users hinders their ability attack the system they are so familiar with. It is worth stressing removing their ability to access the system will likely force those willing to attack to attempt social engineering. One of the most difficult and most important policies to instill and enforce among other users of the system, users must not provide system access or information to former employees. To this end, it is important to also immediately remove the former user from employee databases. Doing so will prevent employees from verifying their position within the company. Employees who have been informed of their impending firing or who have informed their organization of their resignation should be given restricted access to the system. Intrusion detection systems can be used to make sure employees are not trying to access secure information prior to their departure. One company goes so far as to remove the computers of employees the day their contracts are terminated. The company also changes the password of the fired employee's voicemail. Those who give or are given notice of their departure are given restricted access to the system via a temporary password. What's more, the company also reviews recent changes made to the system by former IT employees [24].

Current employees who are not leaving the organization are also a threat. Consequently, the number of employees with overarching access to the system should be severely limited. Ideally, no employee has access to the entire system. A malicious insider with full access to the system could wreak havoc upon the organization. However, it is just as crucial to remember that, while users with carte blanche access may not have malicious intent, any user with malicious intent who gains their access

information can also wreak havoc on the system. As mentioned before, a user with access to the password file for a system can be particularly lethal for the system. In UNIX systems, the use of *shadow passwords* can mitigate this threat, however. When shadow passwords are employed, the passwords in the password file are singularly represented by a placeholder character(s) or flag. All of the other information in the system remains accessible to the application. The encrypted passwords are stored in a separate file that can only be accessed by users with UNIX administrator privileges [28]. It is important to point out, however, that many newer implementations of UNIX-based operating systems use shadow passwords [44]. Thus, this problem is not quite as prevalent on UNIX systems as in years past.

Insider attacks can also be prevented by distributing responsibilities and access to the system. Doing this could force a disgruntled employee into looking for an accomplice. I mentioned previously that 85% of insiders had informed at least one other individual of their illicit exploits. More important is that 22% informed other employees [23]. An employee loyal to the organization can provide warning if a disgruntled employee approaches them.

The process of maintaining the employees' loyalty to the organization is also important. Whenever instituting any new policies, organizations run the risk of disenfranchising their workers when there exists a culture to resist change. Changing a technology policy can do this to a greater degree. A new security policy that is poorly introduced may lead employees to believe that the organization they work for does not trust them. The resulting fallout may not only cause some workers to overlook the illicit

exploits of their co-workers, but could also cause some to become insiders themselves. As a result, clear communication of all technology policies implemented is imperative. Management needs to explain why there have been changes in security measures. Failing to do this will cause imaginations to run wild. Organizations have to show that they trust their workers while still maintaining a healthy level of suspicion. Doing so will inspire trust among them and thus through the organization.

6.3 Case Study – Bucknell University

A university, like other organizations, possesses information that must be kept secure, such as the personal records of teachers and students. However, the university must also facilitate interaction amongst its staff and faculty and not impede their work. The lack of control over the software configurations of computers brought to the campus by students complicates the work of its IT staff. In fact, this predicament is similar to the issues other organizations confront regarding remote access. There are thousands of administrators, faculty, and students, all with varying degrees of technical ability. For these reasons, Bucknell University is an appropriate organization to examine.

6.3.1 Technology Policy

Bucknell University provides every user of the network with what is called the “Appropriate Use Policy” [31], or AUP for short. All users of the system are required to

adhere to the policy. All new users are required to successfully complete a test regarding the Appropriate Use Policy before utilizing the network's Internet. The policy is divided into separate categories:

- “General Guidelines” – This section lists broad codes of conduct regarding utilization of the system. Statements are included requiring the user to not maliciously attack the system and to preserve the security of the system's resources.
- “Appropriate Use” – This section defines what is considered to be acceptable use of the system. For instance, utilizing the system for research or for education is considered valid, as is using the system for administrative purposes. The section also delineates who is considered to be an appropriate user. In this policy, appropriate users include administrators, faculty, students, staff, and dependents of members of those groups.
- “Community Responsibilities” – This section prescribes the responsibilities of users of the system. All users are responsible for having updated virus software on their computer. The university maintains its own machines. The university also provides antivirus software for users that do not already protect their computer with it. Users are also encouraged to refrain from using system services such as email and web pages for disseminating material that can be construed as racist, sexist, intimidating, or otherwise exclusionary or discriminatory. This is of particular note, because court cases have been

based upon offensive or demeaning content sent via email.

- “Ethical Use” – This section describes what is not permitted use of the system. Obviously, users are not allowed to use the system to violate copyright law. Users are also prohibited from creating worms, viruses, and Trojan horses for uses against the system. Users are instructed to not release their password or otherwise compromise their account; and also to not assume control of another user’s account. Finally, users cannot violate any law, regardless of jurisdiction.
- “Network Use” – This section delves into violations regarding the network. Users are restricted to the username and network address ascribed to them. Users’ computers cannot impair the services provided by the system in any way nor are they allowed to individually consume a substantial portion of the resources provided by the system.
- “Voice Mail Use.” – This section applies many of the same policies previously listed to voice mail. Users are instructed to not reveal their password to anyone nor can they assume the identity of another user. Users must again refrain from making offensive or demeaning statements on the voicemail.

Just as important, the policy also addresses the enforcement of these policies. The following policies regarding enforcement are included in the Appropriate Use Policy.

- “Reporting AUP Violations” – Violations of the policy can be reported via an

online form. A copy of this form has been included in the appendix of this document.

- “Response to AUP Violations” – This section announces the measures administrators can take to protect the network from its users. Computers utilized by users can be disconnected if they are disrupting the network. Computers can also be removed from the network for troubleshooting purposes. Violation of the Appropriate Use Policy results in a fine and temporary suspension from network resources, if the computer has to be disconnected.
- “Sanctions” – This section describes the recourses that can be taken against a user who violates the Appropriate Use Policy. The penalties range from educational classes to loss of privileges, reimbursement, suspension or expulsion, and, finally, prosecution for law violations.

6.3.2 System Implementation

Bucknell University’s implementation of its system is just as important as the policy governing it. Due to the necessity for openness in the system, the system approach to malware is reactive in nature. All computers native to the system are protected by antivirus software and are updated remotely. Computers users introduce to the system are required to have antivirus software. A firewall is in place to prevent certain types of files from being passed around the network. However, the two most effective protections

include the use of subnets and a quarantine system. The network is implemented into a variety of subnets. A subnet is a group of access points on the network maintained by a single switch. Computers can be added to the system via these access points. Thus, the subnets are the distribution level of the network. The subnets are not directly interconnected to each other. Instead, each subnet is connected to the core system. The core is divided between two different locations. The data each core uses is backed up at the other core as a safeguard. The use of subnets prevents malware from propagating throughout the network. A worm, for instance, can only infect the other computers on the subnet. The quarantine system also protects the network by autonomously depriving computers believed to be infected with malware of system resources and services. A computer is deemed a threat and quarantined when it has been sending data for 1000 continuous seconds. A quarantined computer is only allowed to receive email. It cannot send email or access the Internet. When the user of a quarantined computer tries to utilize the Internet, a page is displayed informing the user that the machine has been quarantined. The page also describes what actions the user must take to fix his or her computer and have it removed from quarantine [32].

Passwords are assigned to users when they are first introduced to the system. The passwords are randomly-generated, alphanumeric, and eight characters in length. However, only uppercase letters are used. Users can also change their password if they determine theirs is too difficult to remember. However, the university reserves the right to test passwords to make sure they are secure [31]. Users of the system also have a User ID and a PIN number. These are used to access the user's personal information. Both

the User ID and the PIN number are assigned upon a user's introduction to the system and both consist entirely of numbers. Users can change their six-character pin, but not their eight-character User ID.

Social engineering is not a substantial problem, except through email in a technique that has been popularly referred to as phishing. Filters are in place to prevent some email from reaching the users of the system. However, others are able to make it through. The universities IT division, Information Services and Resources, informs the users via email of any widespread emails asking for information that might compromise the users' privacy or the system's security.

6.4 Suggested System Implementation and Technology Policy

The precise configuration of a secure system cannot be specified here. There is no silver-bullet policy and implementation that can protect all systems, or even completely safeguard one system. Despite what recommendations are made here, it is important to remember that they are designed to mitigate, not eliminate, the way users compromise the security of system. Furthermore, system configuration is dependent upon the needs of the organization, the dataflow within it, and the expense the organization is willing to incur. However, recommendations can be made to improve the security of a system, from both policy and implementation perspectives.

Acceptable Use

- Users must utilize the system in a way that furthers the completion of the task(s) assigned to them.
 - *Goal:* To further enhance and develop the organization and to ensure against the misuse of critical resources.
- Users must use the workstation(s) assigned to them.
 - *Goal:* To prevent and recognize users with malicious intent that are in a position to assume another user's identity.
- Users of the system must be authorized by this organization. Guests may obtain temporary login information from the IT department.
 - *Goal:* To prevent the system from being compromised.
- Users are permitted to use email; however the attachments therein will be filtered.
 - *Goal:* To prevent malicious software from infiltrating the system.
- Users cannot access web pages that may be considered derogatory, racist, or sexist.
 - *Goal:* To maintain a work environment that fosters respect and trust.
- Users are responsible and accountable for protecting the information they have access to.
 - *Goal:* Users are an integral part of maintaining secure information.

- Users of the system must agree to abide by the technology policy set forth here.

A signed agreement will be recorded prior to the user's first use of the system.

- *Goal:* To ensure users understand and accept their responsibilities they are given.

Education

- Users new to the system are required to complete a short test regarding the policies associated with the system.
 - *Goal:* To make sure new users fully understand their responsibilities.
- Users will be reminded of the policies pertaining to the security of the system monthly via printed handouts.
 - *Goal:* To encourage users to remain vigilant and aware of threats to the system.
- Users will receive updates regarding change in policy.
 - *Goal:* To inform them of the latest developments and ensure that the users' actions are as secure as possible.
- Users will receive news of incidents that underscore the importance for security.
 - *Goal:* To encourage compliance and vigilance regarding the security policy.
- Users will be able to direct any questions or concerns they have regarding the security of the system to the IT department.
 - *Goal:* To eliminate any confusion regarding the system policies.

Ethics and Legality

- Users are expected to adhere to copyright law.
 - *Goal:* The organization encourages the following of local, state, and federal laws.
- Users are expected to respect other users and their work.
 - *Goal:* To develop trust and encourage the spread of ideas.
- Users are not permitted to access, copy, change, or destroy any file associated with another user [31].
 - *Goal:* So that a user's progress is not impaired and that a user's privacy is ensured.
- Users cannot access the system as another user.
 - *Goal:* To prevent users from being held accountable for another user's exploits.
- Users cannot grant access to the system to anyone else, users or not, regardless of method.
 - *Goal:* To keep the system from being compromised.
- Users cannot release any information, applications, or hardware.
 - *Goal:* To prevent secure information from being compromised.
- Users may not attempt to disable any protective measure in place on the machine, including, but not limited to, the antivirus software, firewalls, and other software aimed at eliminating malicious software from the system or preventing its presence.

- *Goal:* Doing so would compromise the system security and leave it vulnerable to attack.
- Users are not permitted to develop, place, install, or activate any form of malicious software on the equipment assigned to them.
 - *Goal:* Malicious attacks against the system would jeopardize its security.

Information Protection

- All printed documents are to be shredded prior to disposal.
 - *Goal:* To reduce the effectiveness of dumpster diving attacks.
- All disks are to be shredded prior to disposal.
 - *Goal:* To reduce the effectiveness of dumpster diving attacks.
- All electronic media must be degaussed, shredded, or otherwise destroyed beyond repair before disposal.
 - *Goal:* To reduce the effectiveness of dumpster diving attacks.
- All trash is to remain on premises until pickup.
 - *Goal:* Trash publicly disposed of can be taken legally. Keeping it on the premises requires a dumpster diver to commit trespassing.
- Documents cannot be removed from the premises. Remote access may be used to access them.
 - *Goal:* To keep sensitive information from being disseminated.
- Digital media cannot be removed from the premises.
 - *Goal:* To keep sensitive information from being disseminated.

- Data cannot be sent to any individual requesting it until that person has been properly identified as a member of the organization.
 - *Goal:* To prevent social engineering attacks.
- Users may be properly identified via an employee database.
 - *Goal:* To properly identify employees and thwart social engineering attacks.
- Users must submit the daily code requested by the information supplier.
 - *Goal:* To supply an added layer of protection when releasing potentially sensitive data.
- All phones should be equipped with caller ID to ensure that the caller's name and the name assigned to the phone match.
 - *Goal:* To prevent social engineering attacks.

Passwords

- All passwords must be at least 8 characters in length.
 - *Goal:* So as to make passwords more secure.
- All passwords will consist of any combination of three of the following: uppercase letters, lowercase letters, numbers, and symbol characters.
 - *Goal:* So as to make passwords more secure.
- Users are to use mnemonic-based passwords.
 - *Goal:* To increase the security of the passwords.
- Users cannot share their password with any other person under any circumstances.

- *Goal:* To prevent other people from assuming your identity and committing attacks against the system.
- All users without administrator privileges must change their passwords once every two months [1].
 - *Goal:* User accounts that have been compromised will have new passwords and regain their security.
- All users with administrator privileges must change the passwords once a month [1].
 - *Goal:* Compromised administrator accounts can leave a system particularly vulnerable.
- Passwords, new and forgotten, will be issued to the user in person.
 - *Goal:* Presenting passwords in person hinders social engineering attacks and guarantees the proper user receives his or her password.
- Passwords will not be issued to remote users of the system, regardless of the circumstances.
 - *Goal:* Refusing to give passwords to remote users prevents social engineering attacks.
- Passwords cannot be recorded and stored in obvious locations in the workplace.
 - *Goal:* Doing so can enable other users to logon to a system as someone else. Furthermore, an attacker who has infiltrated the premises could use them to make an attack on the system.

- Users cannot change their password to be something recommended to them. A user's password(s) will be generated by him or her alone.
 - *Goal:* Doing so would implement a password that has already been compromised.
- The accounts of former employees must be immediately and permanently removed from the system the day of their departure.
 - *Goal:* To prevent disgruntled employees and other potential attackers from attacking the system.

Privileges

- Users outside of the IT department are not entitled to administrative access to the system.
 - *Goal:* Most users do not need access to the entire system. Granting such access would place the system at unnecessary risk.
- Users outside of the IT department are not entitled to install software on any machine.
 - *Goal:* To prevent malicious software, such as Trojan horses, from being installed upon the system.
- Users outside of the IT department are not entitled to connect any hardware to the system.
 - *Goal:* Hardware modifications can compromise the system if it has not been configured to account for them.

- Users will be restricted to access only the data, applications, and hardware necessary for their job.
 - *Goal:* Doing so limits the damage that can be caused if a user account becomes compromised.
- Users accessing the system remotely will have privileges that are more restrictive than onsite privileges.
 - *Goal:* To restrict the amount of damage that can be incurred by a remotely connected attacker.
- Members of the IT department are entitled to additional privileges as they pertain to their position.
 - *Goal:* Members of the IT department oversee the other users' accounts and thus need certain privileges not available to other users in order to maintain their accounts.
- No less than 2 and no more than 3 members of the IT department can have full administrator privilege.
 - *Goal:* Having only user with full administrator privileges places all of the eggs in one basket so to speak. Limiting the number of users that do have full privilege makes it more difficult for attacker to obtain administrator-level access.
- IT department members with onsite administrative privilege will not have full administrative privileges when remotely accessing the system.
 - *Goal:* An attacker that is able to gain the access information from a user

with administrator access will not be able to control the entire system without being on the premises.

- No single user will be granted exclusive access to any part of the system.
 - *Goal:* To discourage inside attacks and maintain full knowledge of the system.
- Users will lose all privileges promptly following their departure from the organization.
 - *Goal:* To prevent attacks from former employees or those looking to take advantage of their accounts.
- Users given or who have given their two weeks notice will experience further restrictions on their privileges. Users leaving under suspicious circumstances will be more closely scrutinized.
 - *Goal:* To prevent disenfranchised employees from harming the system prior to their departure.

Remote Access

- Users are responsible for the secure operation of the computer from which they are conducting the remote access.
 - *Goal:* Users' responsibility to maintain the security of the information extends beyond the premises.
- Computers used for remote access should be equipped with the antivirus software and personal firewall provided by the company.

- *Goal:* This will protect the user's home computer and thus also the system from attack.
- Users cannot copy files from the system onto their remote computer.
 - *Goal:* To prevent the dissemination of information to a potentially insecure computer.
- Users cannot transfer files from their remote computer onto the system.
 - *Goal:* To prevent files or programs containing malicious code from being placed onto the system.
- Users cannot remotely access the system for personal use.
 - *Goal:* Doing so places the system at an unnecessary, increased risk.
- Users must not remotely access the system from a computer belonging to a network not under full control of the user [33].
 - *Goal:* Accessing the system from an insecure network places the system at risk for attack.

System

- Each user is required to create a password prior to first accessing the system.
 - *Goal:* Prevents the use of temporary passwords, which can compromise the system when used for too long.
- Each user is required to implement a screen saver password.
 - *Goal:* To protect the system from an attacker on the premises.
- All screensavers will be set for 3 minutes.

- *Goal:* To limit the amount of time a vacant workstation can be empty and still be compromised.
- All users with administrative access are also required to utilize boot passwords.
 - *Goal:* Administrative access is targeted by most attackers so an extra level of protection is necessary.
- All equipment and data associated with the system is the property of this organization.
 - *Goal:* The equipment was purchased by the organization and the ideas and information therein were generated on behalf of the organization. As such, the organization can search examine any part of the system.
- Requests for additional hardware and software must be directed to the IT department.
 - *Goal:* To ensure the additional hardware or software does not compromise the system.

Reporting and Response to Violations

- Violations of this policy must be reported to the IT department or manager.
 - *Goal:* So that the compromised system can be repaired in a timely fashion and thus to also the amount of information leaked.
- Violation reports should offer the date, time, nature of the problem, and how the problem originated, if possible.
 - *Goal:* To give the IT department an understanding of what kind of threat

they are confronting. This can dramatically improve the response time of the IT department and its ability to fix the problem.

- The organization maintains the right to remove any machine from the system.
 - *Goal:* To protect the system from a perceived threat.
- The organization retains the right to repair any machine on the system.
 - *Goal:* The equipment and data belongs to the organization.
- The organization reserves the right to use disciplinary action, up to and including termination, when the organization deems it to be appropriate.
 - *Goal:* Instance may arise where a user acts maliciously or ignorantly against the system in such a way that would warrant disciplinary action. Such action would be undertaken to protect the system and its users.

While precise configuration is dependent upon the characteristics of the organization, the following measures should be implemented to further protect the system from human error:

- Antivirus software setup to automatically update and scan the system.
- A firewall to protect the system from harmful external threats and some internal ones.
- Programs that combat adware and spyware.
- An intrusion detection system to monitor highly sensitive information and those who attempt to access it.

- Limit the number of times a password can be changed to three times during the normal life time of a password.
- All passwords should be salted.
- Shadow passwords should be utilized whenever possible.
- The verification process during a login should be slower than one tenth of a second.
- The network should be divided into subnets to restrict the propagating abilities of malware.

It is important to remember that the above policy and implementation suggestions cannot guarantee the absolute security of the system. However, if followed, the above recommendation can improve upon the security of a system. It is also important to instill in the users a sense of trust. Organization should avoid making users overly paranoid about their working environment. Though this may be difficult with the plethora of restrictions above, the users need to be reminded of their importance. The user is the weakest link in the system's security. Users need to be reminded that the organization they work for places its trust them with the information they are privy to. The policy laid out above is not intended to be a straitjacket, but a set of guidelines that will help users maintain system security while still enabling them to perform their jobs.

Chapter Summary

This chapter determined what can be done to mitigate the security threat users pose to systems.

The first section addressed how to account for users who accidentally and unknowingly compromise the system. The section covered how to make users more resilient to attacks involving malware, passwords, and social engineering. A variety of improvements to both the system and to users' habit can be made to improve upon the security of the system.

The second section confronted the problem of malicious insiders. The habits of a malicious insider cannot be improved upon. However, many of the improvements suggested for benign users also help thwart malicious users. The development of trust within an organization, the distribution of responsibilities, and the limitation of privileges all combine to further prevent insider attacks and their effectiveness.

The next section provided a case study of Bucknell University. The Appropriate Use Policy of the university is examined as is, albeit briefly, the orientation of its network. This section served as a sort of precursor to the final section where a policy, based upon the recommendations previously presented, was laid out as were recommendations for the implementation of a more secure system.

Most importantly, people are creative, inventive and as such will find new ways to compromise systems. Organizations must be flexible enough to augment their policies to account for these future breaches.

Concluding Remarks

This thesis confronted the lack of a comprehensive text examining how users can compromise a computer system. The first part of the thesis addressed the plethora of ways users compromise computer systems. The second part described how to account for user behavior via technology policy and system implementation thereby making the system more secure.

The first chapter, "Email Attachments and Downloadable Programs," addressed users' susceptibility to malicious software. It described and classified the various types of malicious software, including worms, viruses and Trojan horses. The various implementations of firewalls and antivirus software were discussed as were their respective weaknesses. The chapter closed with examples of malicious software and highlights the significant threat it poses to computer systems.

"Passwords," the next chapter, highlighted the propensity of users to pick, or be assigned as the case may be, insecure passwords. "Bad" passwords are those selected by most users on their own. They use names, words, birthdates, and are vulnerable to dictionary attacks, among others. "Good" passwords are randomly-generated and are usually assigned to users. However, their often confusing combination of characters forces many users to record their passwords, which they then store in insecure locations. This chapter also confronted the problems created when users utilize the same password across various systems and when users predictably alter their password when a change is required.

The chapter titled “Social Engineering” followed next and described the openness of systems to non-technical modes of attack. The chapter led off by detailing how social engineers are able to establish and use trust to compromise a system. Trailing this was a section describing the various ways a social engineering attack can be carried out. The chapter closed by using the recent scandal with ChoicePoint® as an example of a social engineering attack.

“Malicious insiders,” were the last user-based threat to computer systems investigated in this thesis. Due to common misconceptions about the perpetrators of insider attacks, the chapter first delved into the characteristics of the malicious insider, both popular and factual. Computer systems have an increased vulnerability to insiders, due to their often intimate nature with the system they are attacking. This increased vulnerability is a critical fact and is thus investigated next. The chapter closed with examples of past attackers carried out by malicious insiders and the havoc they wrought upon their organizations.

The second part started with a chapter titled “Ethical Attributes.” This chapter investigated two different ethics common among users: Utilitarianism and Kantian ethics. Utilitarianism bases its decisions upon a cost-benefit analysis of the end results with little regard for how to get there. Kantian ethics focus almost exclusively upon the decision at hand, however in doing so fails to take into account the results of the action taken. The final section debated which mode of ethics is most desirable for the maintenance of a system’s security.

“Addressing User Behavior” was the last chapter and crux of this thesis. The

chapter's first two sections described what steps can be taken to mitigate the effects of users with benign intent and users with malicious intent, respectively. The next section was a case study of the system implemented and policy presented at Bucknell University. The final section described and explains rules that can be implemented in a technology policy that would make a computer system less prone to human error. Different system implementations are also suggested to mitigate the effects of user errors when they do occur.

Despite the thoroughness of the research carried out, this thesis did not address the complications introduced by wireless networks or virtual private networks. Given the time, this thesis would expand into each of these areas and investigate the ways users create system vulnerabilities, such as failing to use WEP encryption over wireless networks. The thesis would also press harder into the implementation of email filters and investigate the feasibility of filtering out executables based upon byte values instead of the file extension attributed to it. Quality Control Circles are small groups of six to ten users that work in a cohesive environment [39]. The use and effectiveness of Quality Control Circles would be considered as a means of maintaining system security through education. Quality Control Circles could also be effective in preventing insider attacks. As a result, Quality Control Circles would be included in an expansion of this thesis.

Bibliography

- [1] Mitnick, Kevin D. and William L. Simon. The Art of Deception: Controlling the Human Element of Security. Indianapolis, IN: Wiley Publishing Inc., © 2002
- [2] ChoicePoint® Update on Fraud Investigation. Internet. Address:
http://www.choicepoint.com/news/statement_0205_1.html. February 21, 2005
- [3] Yan, Jianxin, Alan Blackwell, Ross Anderson, and Alasdair Grant. *The Memorability and Security of Passwords: Some Empirical Results*. Cambridge University Computer Laboratory © 2000
- [4] Schneier, Bruce. Secrets & Lies: Digital Security in a Networked World. New York, NY: John Wiley and Sons, Inc., © 2000
- [5] AskOxford: How many words are there in the English language?. Internet. Address:
<http://www.askoxford.com/asktheexperts/faq/aboutenglish/numberwords?view=uk>. April 5, 2005
- [6] Neumann, Peter G. Computer Related Risks. Reading, MA: Addison-Wesley, © 1995
- [7] Furnell, Steven. Cybercrime: Vandalizing the Information Society. Boston, MA: Addison-Wesley, © 2002
- [8] Anderson, Ross. Security Engineering: A Guide to Building Dependable Distributed Systems. New York, NY: Wiley Computer Publishing, © 2001
- [9] Virus Bulletin : W32/Mydoom.a (W32/MyDoom-A, W32.Novarg.a,) Email

- Worms, Network Worms. Internet. Address:
<http://www.virusbtn.com/resources/malwareDirectory/variants/W32-Mydoom.a.xml>. April 5, 2005
- [10] Forbes.com: Ware Protection. Internet. Address:
http://www.forbes.com/forbes/2004/1004/082_print.html. April 5, 2005
- [11] Press Release: "Creator of 'Melissa' Computer Virus Pleads Guilty in New Jersey to State and Federal Charges". Internet. Address:
<http://www.usdoj.gov/criminal/cybercrime/melissa.htm>. April 5, 2005
- [12] Security Statistics – Virus Statistics. Internet. Address:
<http://www.securitystats.com/virusstats.html>. April 5, 2005
- [13] Feature: The Omega files, 06/26/00. Internet. Address:
<http://www.nwfusion.com/research/2000/0626feat.html>. April 5, 2005
- [14] Disgruntled UBS PaineWebber Employee Charged with Allegedly Unleashing "Logic Bomb" on Company Computers. Internet. Address:
<http://www.usdoj.gov/criminal/cybercrime/duronioIndict.htm>. April 5, 2005
- [15] Vallejo, California Woman Admits to Embezzling More Than \$875,035 (July 28, 2004). Internet. Address:
<http://www.usdoj.gov/criminal/cybercrime/sabathiaPlea.htm>. April 5, 2005
- [16] Spafford, Eugene. *Cyber Terrorism: The New Asymmetric Threat*. Testimony before the House Armed Services Subcommittee on Terrorism, Unconventional Threats and Capabilities July 24, 2003. Internet. Address:
<http://www.acm.org/usacm/PDF/03-07-24spafford.pdf>. April 5, 2005

- [17] Ad-Aware SE Professional – Software – Lavasoft. Internet. Address:
<http://www.lavasoftusa.com/software/adawareprofessional/>. April 5, 2005
- [18] Downloads – The home of Spybot-S&D!. Internet. Address: <http://www.safer-networking.org/en/download/index.html>. April 5, 2005
- [19] What is challenge-response? - A Word Definition From the Webopedia Computer Dictionary. Internet. Address:
http://www.webopedia.com/TERM/C/challenge_response.html. April 21, 2005
- [20] Morehead, Albert and Loy. The New American Webster Handy College Dictionary. New York, NY: Penguin Books, © 1995
- [21] Password cracking – Wikipedia, the free encyclopedia. Internet. Address:
http://en.wikipedia.org/wiki/Password_cracking. April 5, 2005
- [22] Office Space. Dir. Mike Judge. Screenplay by Mike Judge. 20th Century Fox, © 1999
- [23] Randazzo, Ph.D., Marisa Reddy, Michelle Keeney, Ph.D., Eileen Kowalski, National Threat Assessment Center, United States Secret Service, Dawn Cappelli, Andrew Moore, and CERT® Coordination Center. *Insider Threat Study: Illicit Cyber Activity in the Banking and Finance Sector*. U.S. Secret Service and CERT® Coordination Center © 2004
- [24] Dr. Crime's Terminal of Doom - SECURITY - why biggest security risks are inside organization; how guarding against internal threats can protect against external ones; how CIOs balance need to trust workers with efforts to reduce risks - CIO Magazine Jun 1,2002. Internet. Address:

- <http://www.cio.com/archive/060102/doom.html>. April 6, 2005
- [25] Ethical Theory. Internet. Address:
http://www.bluffton.edu/~capstone/fac/ethical_theory.htm. April 7, 2005
- [26] Ethical Principals in Business. Internet. Address:
<http://www.discovery.mala.bc.ca/web/gosselinkd/Ethical%20Principles%20in%20Business.htm>. April 7, 2005
- [27] Granneman, R. Scott. They Know What?!?: Security and Privacy in the Internet Age. Internet. Address:
<http://www.granneman.com/downloads/TheyKnowWhat.pdf>. April 9, 2005
- [28] [Chapter 8] 8.8 Administrator Techniques for Conventional Passwords. Internet. Address: http://www.unix.org.ua/oreilly/networking/puis/ch08_08.htm. April 10, 2005
- [29] Solomon, Frederick. Probability and Stochastic Processes. Prentice-Hall, Inc., © 1987
- [30] Birthday Attack -- from MathWorld. Internet. Address:
<http://mathworld.wolfram.com/BirthdayAttack.html>. April 13, 2005
- [31] Appropriate Use Policy. Internet. Address:
http://www.bucknell.edu/Library_computing/Policies_and_Guidelines/AUP.html April 13, 2005
- [32] Private Communication with Eric Smith, Systems – Integrator and Network Administrator, Bucknell University. April 13, 2005
- [33] Guel, Michele D. *A Short Primer For Developing Security Policies*. The SANS

Institute, © 2001 Internet. Address:

http://www.sans.org/resources/policies/Policy_Primer.pdf.

- [34] CERT Advisory CA-2001-19 "Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL. Internet. Address:
<http://www.cert.org/advisories/CA-2001-19.html>. April 19, 2005
- [35] Private Communication with Felipe Perrone – Assistant Professor of Computer Science, Bucknell University. April 18, 2005
- [36] ChoicePoint – About ChoicePoint/Overview. Internet. Address:
<http://www.choicepoint.com/about/overview.html>. April 19, 2005
- [37] iPlanet Web Server, Enterprise Edition Server-Side JavaScript Guide: Chapter#32;1 JavaScript Overview. Internet. Address:
<http://docs.sun.com/source/816-5930-10/intro.htm>. April 19, 2005
- [38] What is ActiveX control? - A Word Definition From the Webopedia Computer Dictionary. Internet. Address:
http://www.webopedia.com/TERM/A/ActiveX_control.html. April 19, 2005
- [39] Quality Control Circle. Internet. Address:
<http://www.mampu.gov.my/mampu/bi/program/Circulars/DAC0791/DAC0791.htm>. April 21, 2005
- [40] Tutorials – 30: Scripting: Higher-Level Programming for Component-Based Systems. Internet. Address:
<http://www.acm.org/sigs/sigplan/oopsla/oopsla98/ap/tutorial/tutorial.htm>. April 23, 2005

- [41] Document Object Model - Wikipedia, the free encyclopedia. Internet. Address:
http://en.wikipedia.org/wiki/Document_Object_Model. April 23, 2005
- [42] Component Object Model - Wikipedia, the free encyclopedia. Internet. Address:
http://en.wikipedia.org/wiki/Active_X April 23, 2005
- [43] Macro Viruses Internet. Address:
<http://www3.ca.com/securityadvisor/newsinfo/collateral.aspx?cid=33338>. May
2, 2005
- [44] What are shadow passwords? Internet. Address:
<http://kb.indiana.edu/data/aezz.html?cust=976747.33395.30>. May 2, 2005