PROACTIVE MALWARE PREVENTION

GARRETT PICKERING
ALIJOHN GHASSEMLOUEI
TREY WILSON
LAUREN HARGIS
VICTOR TEISSLER

2009 UNIVERSITY OF ADVANCING TECHNOLOGY

Proactive Malware Prevention Software

Thirteenth Draft

Lauren Hargis, Alijohn Ghassemlouei, Garrett Pickering, Trey Wilson, Victor Teissler

University of Advancing Technology

Table of Contents

Abstract

This dissertation considers current antivirus methodologies, including any obstacles that may occurring during development. The primary objective was to utilize existing technologies to create a new and innovative methodology for detecting and preventing malware from executing on systems. The outcome was the Proactive Malware Prevention System: a ground-breaking business tool created to provide security, control and ease of use for all parties involved. While security was the main focus of this project, it evolved to provide an extensible tool for administrators to utilize for rapid deployment of software packages and provided an easy to use interface for employees to access data based on their needs. The central focus for the development of this tool was to successfully deploy the tool within a business environment and also to evaluate the performance and usability of the tool itself. Of the individual components within the project, it was crucial to consider the various different aspects of the programming, network communications, and user interface methodologies.

Introduction

The industry standard best practices for antivirus protection are severely flawed; these programs are limited to locating malware through pattern recognition using predefined malware libraries. While this protection method will be more than enough for home users, businesses have more demands that need to be met. Within the business environment, it is not a smart choice to take risks with the IT infrastructure; it is only a matter of time until the protection systems implemented will be penetrated. Risk management and mitigation are essential when managing the IT infrastructure within a live business environment; thus, outdated methods for antivirus protection are in dire need of a revitalization.

The business world is in need of a new form of host protection: operational whitelisting is the best approach for overcoming the flaws of current antivirus protection because a proactive approach eliminates many issues before the system or user can be affected. Network administrators can certify which programs are allowed to run on which machines and create images of programs that can be pushed out to multiple clients at once, thus creating a more efficient, secure, and manageable network.

Within the project's implementation, for a program to execute on a client machine, that particular machine needs to be authenticated properly with the server. Only after it has passed multifactor authentication will the client be allowed to run programs that the network administrators have chosen.

The white listing method is extremely effective because it has a specific list of approved programs that can be run on client machines, which cuts down on misuse of company machines and adds another level of protection against malware. For example, the accounting department machines require different pieces of software than the human resources department; each department could be assigned a different list of approved programs on a group-policy or even a user-policy basis. This method will eliminate unnecessary programs being installed and run on live business networks unless approved by the network staff.

Statement of Problem

Most companies must rely heavily on their IT infrastructure. If the infrastructure is not properly secured, then it can be susceptible to malware, which could be disastrous for both the business and all other involved parties leading to increased expenses. Studies show that about 70% of companies with at least 100 computers have active malware on their network (nanoadmin, 2007). Antivirus programs can only detect known forms of malware and require regular updates to remain effective. Antivirus agents cannot detect many forms of polymorphic viruses because they are always changing and; therefore, always new (Kruegel, Kirda, Mutz, Robertson, & Vigna, 2005). The proof of this issue is that most companies have standard antivirus programs installed to protect their networks, and the problem still remains. There needs to be a program that not only defends the hosts within the network but also deters or prevents any form of malware.

Research Question

How would disallowing all forms of non-verified programs be a more effective method of preventing malware from compromising [networked] computers than traditional signature-based antivirus solutions?

Hypothesis

Systems implementing a proactive whitelisting approach, which deny all forms of non-verified programs, will provide a more effective method of preventing malware from compromising networked computers in comparison to traditional antivirus software.

Project Plan

The project can only be completed when a proof of concept has been created. The first step is to consider the issues that the business community is having with antivirus solutions and application control. From collected information and resources, this is a feasible project due to the knowledge and motivation the team possesses. The information that needs to be researched consists of current methods of antivirus protection and the programming elements that are unfamiliar to the team.

With regards to the project scope, it is important to clarify the average length of time for an average research project, which can take upwards of three to four years. However, a team of four individuals will handle this project, so the scope will entail a three to four semester development period. Within this time period, there will be four stages that will be completed. The current estimated project time frame is expected to be from September 8, 2008 to December 14, 2009; within this time, research will be done, information will be collected, and a product created.

The first task is to identify and analyze the current code and documentation, establish the needs of the project, organize our thoughts, and research information in preparation for stage two. Stage two (the programming stage) is divided into three subcategories: client, server, and management console. These three subcategories will then be distributed to specific team members who will have a deadline to meet for particular tasks within that stage.

Stage three will entail testing the developed program for inconsistencies and data collection. Shortly thereafter, stage four will bring everything together; the programmers will work on finalizing the code, and integrates the graphical user interface while the rest of the team prepares the marketing materials.

During the development process, a lab environment will be created to provide the programmers a place to test the program and fine-tune the finished product. This environment will include three dedicated machines: a Windows 2003 server, Windows XP client, and a Debian server. After the development is complete, another Windows XP client machine will be added to the environment, and the program can be tested in a multi-client environment. When this has been completed, the durability of the program will be tested. One client will use the developed program, and the other client will use an enterprise level antivirus solution. These client machines will then systematically be tested against many different types of viruses, trojans, and other various types of malware in order to compare the effectiveness of the two methods.

Key Terms

Antivirus – A software program designed to identify and remove a known or potential computer virus.

Backdoor – A means of access to a computer program that bypasses security mechanisms.

Blacklisting – A basic access control mechanism that allows everyone access, except for the members of the black list (i.e. list of denied accesses).

Client – A workstation on a network that gains access to central data files, programs, and peripheral devices through a server.

Code – The symbolic arrangement of statements or instructions in a computer program in which letters, digits, etc. are represented as binary numbers; the set of instructions.

Contaminants -

DMZ – [Demilitarized Zone] A physical or logical subnetwork that contains and exposes an organization's external services to a larger, untrusted network, usually the Internet.

DOS Attack – Abbreviation for Denial-Of-Service attack. This abbreviation is most often used to describe attempts to shut down networks through massive digital flooding.

Hook in – [Process Hooking Reference]

Images – The state of a computer or software system stored in some non-volatile form. The form of storage is often a file.

IT – [Information Technology] The development, installation, and implementation of computer systems and applications.

Malware – Malicious computer software that interferes with normal computer functions or sends personal data about the user to unauthorized parties over the Internet.

Polymorphic Virus – A virus that changes its virus signature every time it replicates and infects a new file in order to keep from being detected by an antivirus program.

Program – The precise sequence of instructions enabling a computer to solve a problem.

RAT – [Remote Access Trojan] – Trojan horse that provides the unauthorized third party, with a backdoor into the infected system. This backdoor allows the hacker to snoop your system, use your infected system to launch a zombie attack, or even run malicious code.

Signature –

Software – The programs used to direct the operation of a computer as well as documentation giving instructions on how to use them

System –

Trojan – A class of computer threats that appears to perform a desirable function but in fact performs undisclosed malicious functions that allow unauthorized access to the host machine.

User – A person who uses a computer.

Virus – A segment of self-replicating code planted illegally in a computer program often to damage or shut down a system or network.

Worm – A self-replicating computer program. It uses a network to send copies of itself to other nodes (computers on the network), and it may do so without any user intervention. Unlike a virus, it does not need to attach itself to an existing program

Whitelist – A list or compilation identifying persons or organizations that are accepted, recognized, or privileged.

Chapter II – Literature Review

The modern computer virus is code that has been written with malicious intent which infects a computer only when there is user complicity. The user has to execute the infected program in order for it to take action. A worm is similar to a virus, but the worm can spread without user intervention. Some common ways for worms to spread are using a person's email list to send itself to the persons contacts and finding exploitable vulnerabilities in an operating system or service. A Trojan is a virus that is not self-replicating. A Trojan is typically sent to a person and resembles a safe program, but, when executed, it infects the computer. Another form of Trojan is a RAT, or remote administrative tool, which can be used to control a computer from across the internet (Notenboom 2008; pdesigns, 2006).

Computer viruses, worms, Trojans, and any other programs that have malicious intent are called malware (Moir, 2003). Antivirus programs try to keep a computer from becoming infected with malware by detecting known versions of malware on the system and eliminating them. The antivirus software detects malware by scanning files that have been cataloged as malicious. Some antivirus programs also detect processes that are being issued by each program running on the computer and compare them to the process patterns of malware (Antivirus, 2008). No matter the detection method, the cataloging method is always the same: signatures. Antivirus companies generate signatures for known malicious software using their own proprietary algorithms. Next, they generate a signature for every file on the computer and compare these to the malicious signatures. The antivirus programs typically update through the internet, so antivirus programs always have the most current malware signatures cataloged

(Berger, 2002).

Signature-based malware detection has proven itself an ineffective method for malware prevention; local and network resources are squandered on continuous patching and updating of definitions. The primary problem with this type of detection is that antivirus programs can only detect known malware. There were an estimated five new computer viruses created a day in 2002, and that number has only increased ever since (Antivirus 2008). One study showed that modified versions of a virus are often detected within days of the virus's initial release. For example, the day after the Love virus was released, 27 modified versions of the virus were released from different sources around the world (Essortment, 2002). This is problematic because antivirus programs have to capture a copy of each variation to completely secure a system. Also, if someone were to create a virus and spread it across a private network, signatures could not be generated and distributed to counteract the outbreak until an individual notified the antivirus agencies (Antivirus, 2008). Another classification of viruses that are an emerging problem are polymorphic viruses. They are viruses that change their code every time they infect another computer. The problem with this type of virus is the antivirus will not detect the mutated virus because it has a different pattern in the code. Polymorphic viruses are extremely dangerous because they cause problems similar to those posed by programmers modifying existing viruses (Kruegel, Kirda, Mutz, Robertson, & Vigna, 2005).

Most corporations use the best antivirus programs available because they have so much at stake, but their protection is not complete (Physorg, 2007). Polymorphic viruses and new malware coming out every day can still infect even the protected computer. There are several reasons companies become infected with malware, but, no matter the reason, it costs money. One report stated nearly half of companies worldwide have suffered some form of business disruption due to malware (Millman, 2007). The same report stated that “26% of enterprises reported that confidential information had been compromised as a result of spyware” (Millman, 2007). A company with 14,000 workstations lost $108,000 because of the infection (Millman, 2007). Malware is the most expensive electronic incident that can affect a company. The second most expensive electronic disaster is a DOS, or denial of service attack. A DOS occurs when one or more computers send so much data flowing at a web site or network that it cannot send any information (Roebuck, 2005). One of the worst viruses, MyDoom, carried a payload that was a DOS attack so anyone infected with the virus would participate in the DOS. (Hypponen, Tocheva, & Rautiainen, 2004).

Some of the primary reasons that companies become infected with malware are malicious employees, untrained employees, untrained IT personnel, and insecure networks. Disgruntled employees are among the most common and most damaging vectors through which malware is introduced (Espiner, 2007). Untrained employees are not far behind. One study showed that president’s, vice president’s, and CEO’s were the most likely to download and install software even when the purpose of the software was not clear to them. Another study showed that for a free pen, 90% of employees were willing to write down their password on a survey. Two thirds of those employees admitted that they used the same password for everything including their bank accounts (Leyden, 2003). Intention does not always correspond with action. For example, the president of a company quickly said that he would not distribute his password. However, when asked what it was, he said his daughter's name. When asked what that was, he replied, "Tasmin, oh wait" (Leyden, 2003). An untrained IT department can be equally devastating because it can lead to an insecure network. There have been several cases where public web servers have been on the same network as accounting computers and other sensitive information instead of a DMZ (demilitarized zone), which can be disastrous (Schinder, 2004).

Since a single virus infection can cost a company millions of dollars, there is a rising need to prevent malware infections of any form. Current protective technologies are constantly improving, but malware is improving just as quickly. Antivirus software has many flaws but is currently the only solution. Another possible solution would be analogous to how the human body fights pathogens. Current antivirus solutions search for known malware but cannot detect unknown malware; in contrast, the human body knows only what is supposed to exist. If anything contained in the body is a foreign element, the body eradicates the contaminant without hesitation (Hunt, 2006; Anitei, 2007).

Chapter III - Methodology

Every day, computer users around the globe are being infected by software designed with malicious intent. Among these users, businesses of varying sizes lose millions of dollars every year. Antivirus solutions exist in order to help prevent disastrous results for businesses and home users alike. However, the construction of these solutions neglects certain key elements, which leaves room for improvement.

Antivirus products on the market today use algorithms that are designed to detect malicious software that is already present on the computer and attempts to prevent its execution. Some software solutions proactively scan for anything harmful before it has a chance to run, but this only detects known malware. This often means the malware will not be removed until after the damage has been done. Too often, a virus is released and causes millions of dollars worth in damages before the antivirus databases can be updated with proper signatures.

Through the study of the human body, biologists have learned that the body does not identify malicious germs, viruses, or bacteria; it only identifies what is not supposed to be in the body. Anything that the body does not recognize is therefore unwanted and eventually removed. From this thinking, a new method is born.

This ideology can easily be transported to everyday computer systems. Instead of trying to hunt down and eliminate bad programs, protective software simply knows what is supposed to exist on a system and treats everything else as unwanted. For example, running a game such as Solitaire on a system that does not have it listed as acceptable software will not work.

The project objective is to create an application that can be told what software is acceptable and allow only that approved software to execute. System administrators will easily be able to add or remove software by approving it and pushing it out to the users. The target audience is that of a business environment, mainly due to the fact that the infrastructure is already available and configured accordingly. Home users lack the proper client/server environment necessary for the current release; however, future versions will allow for single-machine usage.

Once implemented, this method will be tested alongside the standard antivirus method currently utilized by the business community. The two methods will be installed side-by-side on networked systems. They will undergo extensive testing for reliability. After populating the database with the approved applications, many forms of malware will be installed and executed to determine which method handles the malicious code more effectively (as measured by the infection rate).

Based on preliminary investigation, our implementation should outperform the old method that uses detection algorithms. The new method will not need to retroactively scan for harmful material; instead, it will pro-actively prevent all forms of malware from executing.

Problem Overview

In a business environment, keeping the company network and computers secure is a top priority, especially when the work to be protected is stored on various devices throughout the network. Authors of malicious software can potentially do much harm to a system or networked systems. The intent of malware varies; there is no one specific goal. Whatever the goal, the software can be considered malicious and unwanted. Common antivirus solutions work retroactively, meaning they will only detect malicious software after it has been introduced into the system.

The purpose of the malware prevention software is to do just as stated: prevent malicious software of all types from being able to use the resources of a computer system. Using a specific set of rules and guidelines, it is possible to secure a system and inform the computer about which software should be allowed to run. The target environment is the Windows-based platform because it is the environment in which malicious software is most prevalent due to the exploitative nature that the platform provides.

Research Question

The investigation of the research question can be distilled into three distinct parts. The first part of this research will explore how current antivirus solutions operate in order to better understand their shortcomings. The next part will determine how a better solution can be developed within the constraints of a Microsoft Windows environment by implementing a proactive approach. The final consideration will be to determine the effectiveness of the proactive malware prevention software. All of the parts presented are sequential, meaning one cannot be researched without having properly concluded those preceding.

Research Hypothesis

Systems implementing a proactive whitelisting approach, which deny all forms of non-verified programs, will provide a more effective method of preventing malware from compromising networked computers in comparison to traditional antivirus software.

Population and Sample

Choosing an appropriate population is essential for the research project. Since the antivirus solution targets networked business environments, it is apparent that the population should be based on small to medium sized businesses. The amount of people needed to conduct thorough research will vary; however, a set of five individuals should provide sufficient data.

Table 3.1 - *Business Size Enumeration*

Small Business	Medium Business	Large Business
Less than 100 employees	Less than 500 employees	More than 1000 employees

The targeted sampling group will consist of network administrators who have worked within small to medium sized businesses, have advanced knowledge in various different fields within information technology, and have a minimum of five years experience in their respective areas of expertise. In addition to the administrators who will be testing out the server-side aspect, there will also be end users who will test the client. The individuals dealing with the client will not need any applicable experience other than basic computer knowledge such as word processing and general data entry.

There are two main categories for the population and sample. The first category consists of the three main components of the solution: the clients, the server, and the management console. Each individual component will have a built-in mechanism for providing useful statistics that can be collected and analyzed at any time. In the data, the software will record the number of allowed processes, the number of denied processes, the names of the denied processes, how many times the denied processes tried to run, the amount of time the produced software was used, the total number of clients being used in that particular network environment, and any other data that will be helpful. Many more statistics will be provided as development continues.

The second part of the population will consist of a simple survey, which is built into the program for easy access. Due to the scope and time constraints of the project, only a small business environment will be constructed in order to test. The users and administrators of the test environment will be asked to use the software and fill out a simple survey describing the experience and usefulness of the software. Other critical information includes usability, aesthetics, and a place to put optional information or general suggestions.

The two main populations here will provide enough information to determine the direction in which development should continue and provide further data on the usefulness of the “whitelist” methodology. Many more options will be explored as development continues, so the current population and sample is subject to change. Additional testing may be required to gather sufficient data to properly guide the project.

Data Collection and Instrumentation

Data for the project will be collected by a means of software evaluations in a business environment. In order to prove this project functions within a business environment as described, it must be tested within one. There will be two phases of testing and collection.

In the software evaluation phase, many aspects of the software will be tested for technical competency. Attributes such as usability, comfort, and practicality in a business environment will not be tested in this phase. The software should be able to function as programmed within the constraints and requirements of a typical business environment. It is designed to accommodate the number of users that may be found in both small and large businesses. During this process, the software will log and maintain various technical details that can later be analyzed in order to evaluate the performance of the software solution. If any optimizations for performance are necessary, this phase is where they will be identified.

The second phase will test the usability of the software. This can be done with a survey of participating testers, which can easily be built into the program. During this phase, a test business environment will be constructed in order to best simulate the intended operating conditions. The software solution will be subjected to a variety of tests that will measure usability, aesthetics, and the practicality of the solution in a business environment in comparison to standard antivirus competitors.

In order to progress through the second phase, a networked environment containing two computer systems will be used. On the first system, the developed program will be installed, and an off the shelf antivirus product will be installed on the second system. Then, several types of malware will be introduced to the systems. Data will be collected on the different types of malware, detection time, and neutralization time during the testing period. Estimations will be made on potential damage that could have been done to the mock business environments.

Known viruses will be tested on the computers because this is the regular antivirus method’s strong point. If the experimental software prevents the same amount (or more) of viruses as the standard antivirus, then that evaluation will be considered a success.

New and polymorphic viruses will be tested on the computers as well. The old method needs to know what malware looks like in order to prevent malware from infecting the host computer whereas the new method will prevent the viruses without having to consult a signature library.

Data Analysis

Data analysis can be separated into three stages: reporting, environment comparison, and survey analyses will be essential to this project to provide sufficient information to improve the program and show statistical data on its effectiveness.

Within the reporting stage, any data that has been collected throughout the project will be compiled into a report; this report will have three categories: allowed processes, denied processes, and resource usage data. A breakdown of each section is necessary due to overlapping data in many different fields; only after the data has been organized can the data be compared and properly analyzed. Due to the nature of this project, it is imperative that the analyzed data provide an accurate study of the program and meaningful feedback for the development team in regards to efficiency and overall security of the application.

In the allowed processes category of the reporting stage the factors to be analyzed will include the number of programs executed, hourly usage, and all errors. The denied processes category will examine the name, time stamp, source, and number of attempts related to each incident flagged as a denied process, as well as any unauthorized access to protected data. The final category involves the analysis of resource usage across all clients authenticated to the server. The number of sessions, session time, and bandwidth usage will also be taken into consideration. After the data from the three categories has been separated, statistics regarding the efficiency and effectiveness of the application can be derived so that the development team can make adjustments to improve the quality of the application.

The second stage is concerned with the comparison betwixt traditional antivirus solutions and the proactive whitelisting solution. Similar to the categories in the previous stage, the three main categories of comparison within this stage will be detection, mis-detection, and resource usage. All solutions will be rated on a scale of 1 through 10 (10 being the best and 1 being the worst), the scaling being based off of the accuracy of objectives completed. Out of the programs tested, the program that yields the highest rating based off of the requirements of that assessment will be identified as the better product within that test.

The first testing category within the second stage will rank applications in rapid malware identification, proper malware identification, proper neutralization, and proper reporting and logging. The second testing category will rank applications based on number of improper identifications, neutralization method used, logging, location of infection, and additional information regarding the origin of the malicious software. The final testing category consists of local and network usage reports which will be included in the overall score. The final results will be based off of how well the requirements were met.

The end-user feedback survey will provide developers with more insight as to how usability can be improved. The survey will cover the user’s general experience with the application, the user’s opinion on various facets of the program (such as usefulness, usability, aesthetics, security, and efficiency), and any additional suggestions.

Summary

Selection of an appropriate sample is essential; the target sample population will be that of small to medium sized businesses. This population will take into consideration subject matter experts in the fields of both network and system administration to provide appropriate feedback for further development.

The remainder of the testing will be dedicated to testing each component separately and ensuring that the standards originally set by the development team are met. Data collection will continue during testing. The server, client and management console, will record any errors and performance data so that it may be evaluated at a later time.

The first phase of data collection is strictly limited to testing the developed software. The developers set standards as to the functionality of the finalized program, such as the performance logging standard (logging functionality built into each component). These standards are in place to enable the development team to produce a product that can be successfully sold within a business environment. If these standards are not met, the project must be re-evaluated.

After the data collection phase has been completed, the analysis of the collected data can begin. Data analysis is separated into three stages. The first stage is to analyze the performance of the product in regards to detection accuracy, allowed processes, and any other applicable information. The second stage is an analysis of the product’s performance in comparison to competing products that utilize existing blacklist methodologies. The final stage is an analysis of the surveys which poll the overall usability and appearance of the graphical user

interface. Each stage includes various tests so the to validate the effectiveness of the program as well as reveal areas for improvement.

Chapter IV - Results & Discussions

The purpose of the research is to demonstrate that the method of denying all non-approved applications is more effective than blocking known malware. Through a series of rigorous tests, the developed program will be compared to traditional antivirus solutions in a closed environment to reveal which solution will prevent the most infections. The data collected will consist primarily of the logs and reporting of the client and server systems within the testing environment. However, additional feedback will be collected from the administrators and end users in order to improve the functionality and efficiency of the developed program.

Analysis Type

The purpose of the study will be to show that our solution will serve as a better option for preventing malware than traditional antivirus solutions. There are two types of data collection methodologies that will be employed within this project: quantitative and qualitative. Through a quantitative analysis it will be clear that the software will effectively work in place of traditional antivirus solutions. A qualitative examination allows experts to voice their opinions regarding the efficiency and usefulness of the application.

The whitelisting software will be configured within a test environment and tested thoroughly to see the degree of the contamination introduced by the malware and the resulting damage if the systems have been compromised. The environment will be exposed to common forms of malware. The purpose behind this is to simulate not only the forms of malware that a company will naturally run into but also the forms of malware that a disgruntled employee could create and introduce into a network. All successful, failed, and denied infections will be documented for statistical analysis. The same tests will also be completed using commercially available antivirus software in order to compare the results.

IT professionals with real world experience in medium to large corporate networks will have access to the software for evaluation purposes and will complete a short survey after a two-week usage period; interviews will also be conducted to get more feedback to further enhance and provide ideas and real world implementation details. This will provide valuable information that is part of qualitative analysis to show that the project is able to meet the needs of both end users and that of the IT professionals while improving security and efficiency of the network.

Data Collection Methodology

The goal of the malware prevention software is to prevent as many computer contaminants as possible; thus, the primary analysis method for this project is a quantitative analysis. This method was chosen because the focus is to protect the system from infection; a qualitative approach will be introduced when collecting user feedback. Ideally, the program will deter any malware, and there will be no infections during the testing phase. Current commercial antivirus solutions will be exposed to new and known viruses, and the results will be recorded and analyzed. If the experimental software prevents the same amount (or more) of malware as the standard antivirus solutions, then the testing will be considered a success. Modifications and improvements will be made accordingly to make the security of the program. The data collection methods in use will be a combination of observation and sampling.

Research Question and Hypothesis

An analysis of the data compared to the original research question should yield some interesting results. Is disallowing all forms of non-verified programs a more effective method of preventing malware from compromising computers in a corporate network than traditional signature-based antivirus solutions? The type of data collection that we will be using should be able to prove that the whitelisting method works to a much higher degree of success.

Systems implementing a proactive whitelisting approach, which denies all forms of non-verified programs, will provide a more effective method of preventing malware from compromising networked computers in comparison to traditional antivirus software. If the data collection shows a higher infection rate on the system using the whitelisting method, then the conclusion will be that the hypothesis was incorrect. Otherwise the system will have a lower infection rate than the system using the standard antivirus method and the hypothesis will be true.

Chapter V – Conclusion & Recommendations

After rounds of testing, IT professional review and recommendations the project has come a long way. As with all new ideas and concepts, there are still many refinements to be made. Testing has shown that the project can be more effective than standard antivirus solutions in stopping malware from executing on the client computer if used with proper whitelists. It could in the future replace, or at least compliment commercial antivirus solutions. With only tested and approved programs being allowed to run, IT costs will be reduced as less time will be spent maintaining systems that were modified against IT policy by users.

Summary

The purpose of the research was to see if disallowing all forms of non-verified programs is a more effective method of preventing malware from compromising computers in a network based environment than traditional signature-based antivirus solutions. Most antivirus software uses a blacklisting method that disallows applications known to be malicious from running. A whitelisting method disallows all applications from running except applications that have explicit permission. This project has proven that whitelisting is a viable way to stop malware from compromising systems. The collected data showed that the whitelisting method even prevented unidentified malware from executing and causing harm. The project was also thoroughly reviewed by IT professionals who were enthusiastic about the implications of the project and were looking forward to being able to do trials on live networks. Although the studies were limited in scope, we are able to conclude that whitelisting will make managing a computer network environment simpler and secure it against both unknown and known malware attacks.

Findings

The program was tested as described in Chapter IV and the results were conclusive as well as positive. The systems were tested using ten publicly available viruses that are known for high infection rates, and were available to all antivirus vendors software involved in the test. Five new viruses were also written specifically to test the various systems on their accuracy and consistency in the face of a new and unknown threat.

The first test environment, secured by commercially available antivirus software was protected against known threats, but unknown threats were allowed to execute without detection. The ten publicly available viruses and malware were successfully detected and protected against, either by notifying a user or by quarantining and allowing the user to remove the malware. The five viruses that were written to test new threats were not caught or quarantined, and the user was not notified that malware was introduced to the system. One of the viruses disabled the protection software allowing for further infections by other malware. After each piece of malware was introduced to the system, it was reset to its previous state to make sure that no previous malware would affect the results of the next tested malware.

The secondary test environment ran the Proactive Malware Prevention Software, and the same testing procedure followed. The ten publicly available pieces of malware were unable to execute; the application logged the attempts and notified the user that the software was not allowed to be executed because of security policies. One flaw that was present in the application was the ability to execute malicious interpreted code using whitelisted interpreters.

Examples include the Microsoft Visual Basic interpreter and the Python interpreter. We removed the interpreters from the whitelisting and these programs were now successfully were no longer allowed to execute; and the attempts were logged. More fine-grained access control may be implemented as a result. The five viruses that were written as test cases all were logged and were unable to execute within the whitelisting application environment. The tests proved successful in that the whitelisting application is more consistent and able to block both known and unknown threats. After each attempt to run a piece of malware the system was restored to a known uncontaminated state to not have any adverse affects on the testing of another piece of malware.

The testing concluded with describing the project and what it could accomplish to various IT professionals. Regrettably the team was unable to let them test the application within their own environments as at the time of the review it was not yet fully implemented. The professionals were chosen because of their real world experience running computer networks ranging from small to large businesses. They were then asked for their feedback on the system. The reviews that were received were positive and gave feedback regarding how the system would be implemented within various IT organizations. There were some questions raised regarding the capabilities of the system, and whether or not the team could fully deliver those features. The professionals agreed to come back for another review when the system has reached a stage in its development where they would be able to test it in their environments.

Conclusions

The goal or purpose of the research was to find out if disallowing all forms of non-verified programs is a more effective method of preventing malware from compromising computers in a network environment than traditional signature-based antivirus solutions. The final conclusion is that it is a more secure method. The software that has been developed left a 0% infection rate in the test environment. The software is being considered a success; the tests were conclusive, and the IT professionals selected to review the software were impressed and looked forward to the final completed project. The software is innovative, original and well thought out with research to support the claims presented.

The idea of going against standard antivirus methodology is completely appropriate when examined specifically for a business environment. The design of this software implemented in a small, medium, or large business environment should increase the productivity of the IT staff as well as cut back on down time for the rest of the employees in the company.

Implications

The issue that was raised while testing and reviewing the software is that the IT professionals were not able to test the software in a working environment. Instead of being able to review the project as a whole, only the malware prevention functionality was able to be tested by the professionals. After project completion the IT professionals agreed to return again for a second review to test the software in its entirety and give updated feedback.

Future Research

Because the scope of this project goes beyond a simple antivirus method, more comprehensive research should be done into other possible uses of the application. This product can track what applications are in use by specific employees at exact times as well as statistical network and software analysis, meaning employees can be tracked in their use of company resources to discover and eliminate inappropriate behavior.

This project uses whitelisting to allow only certain applications to execute; the same thing could be used to set specific policies regarding employee use of various “time-wasting” computer programs such as Solitaire and Minesweeper. The project could potentially contain a scheduling component that would allow and disallow certain applications at certain times and or limit the time they are used by closing the application when a time limit is exceeded. For example, Google has a 20% rule, which allows employees to spend a portion of their time however they wish. The scheduling component within the application would allow companies to implement custom time management methodologies to maximize employee efficiency.

Final Summary

The purpose of the research is to demonstrate that the method of denying all non-approved applications is more effective than blocking known malware. Through a series of rigorous tests, the developed program has been compared to traditional antivirus solutions in a closed environment to reveal that the hypothesis has proven accurate. The data collected from the malware infection tests has proven that a whitelisting method in an office environment can be much more beneficial. The feedback from the IT professionals was positive and further test will be conducted to produce further feedback and ultimately shape the future of this project.

The test environment that was secured by commercially available antivirus software was protected against known threats, but unknown threats were allowed to execute without detection. One of the viruses even disabled the protection software allowing for further infections by other malware. This is not sufficient for companies that are spending millions of dollars in malware damage prevention and recovery. If the whitelisting method can ultimately save companies money while maintaining a more secure environment than it has outdone the current standard.

The test environment that was secured by the whitelisting method protected against all forms of malware tested and even informed the administrators of the potential threat. The tests were successful in proving that the whitelisting application is more consistent and able to block both known and unknown threats based on whether or not they are allowed to execute in the first place. This system, when implemented correctly, will not need the constant maintenance that many antivirus solutions require because the methodology itself in simpler in design. This is yet another positive step in the right direction.

The IT professionals that examined the product have each given positive feedback as well as imposed import questions. The questions that have been raised that were not immediately answered are being looked into and several of them have already been tested and, if needed, solutions have been implemented. The project has grown from exposure not only from positive support but from critique as well. The finished product will go far beyond the original scope because of the ever expanding ideas and opportunities.

The final conclusion on the project is that it is a success. From the start there were questions of how the software would actively block all form of malware, how the client-server model would work, how the graphical user interface would be received, and how it would all fit together. These questions have been answered in the form of a finished project that produces results far above the prescience set by the antivirus methods currently used in the field. By going against the norm and using the exact opposite of the current standard method a product has been created that could potentially secure company networks far greater than their current standing.

The project has come very far and will go further still. From conception to current design to future plans the project has made movements in the right direction and future implications of this project are limitless.

References

Anitei, S. (2007, April 18). How do we fight against virus attack?. Retrieved August 3, 2008, from Softpedia Website: http://news.softpedia.com/news/How-Do-Our-Bodies-FIght-Against-Viruses-039-Attack-52395.shtml

Antivirus World. (n.d.). Retrieved August 3, 2008, from How does anti-virus software work? Website: http://www.antivirusworld.com/articles/antivirus.php

Berger, S. (2002, July 18). How to update your anti-virus software. Retrieved August 3, 2008, from AARP.org Website: http://www.aarp.org/learntech/computers/howto/a2002-07-18-upgradevirus.html

Espiner, T. (2007, July 27). Locating the real threats to corporate security. Retrieved August 4, 2008, from ZDNet.co.uk Website: http://resources.zdnet.co.uk/articles/features/0,1000002000,39287839,00.htm

Essortment. (2002). Computer virus prevention. Retrieved August 4, 2008, from Essortment Website: http://www.essortment.com/all/computervirusp_pfw.htm

Franz, M. (Ed.). (2004). *Malware: Fighting malicious code*. Upper Saddle River, NJ: Pearson Education, Inc

Hunt, M. (2006). Interferon. Retrieved July 15, 2008, from http://pathmicro.med.sc.edu/mhunt/interferon.htm

Hypponen, M., Tocheva, K., & Rautiainen, S., (2004, January 28). F-secure virus descriptions : Mydoom. Retrieved August 4, 2008, from F-Secure Website: http://www.f-secure.com/v-descs/novarg.shtml

Jowitt, T. (2008, July 18). Study finds huge rise in malware this year. Retrieved August 4, 2008, from PC World – Business Center Website: http://www.pcworld.com/businesscenter/article/148609/study_finds_huge_rise_in_malware_this_year.html

Kruegel, C., Kirda, E., Mutz, D., Robertson, W., & Vigna, G. (2005). Polymorphic worm detection using structural information of executables. *Proceedings of the International Symposium on Recent Advances in Intrusion Detection, USA*. 207-226.

Leyden, J. (2003, April 18). Office workers give away passwords for a cheap pen. Retrieved August 4, 2008, from The Register Website: http://www.theregister.co.uk/2003/04/18/office_workers_give_away_passwords/

McAfee. (2004, November). Effective management of anti-virus and security solutions for small businesses. Retrieved August 3, 2008, from http://64.233.169.104/search?q=cache:2-IlewPb3L4J:www.mcafee.com/us/local_content/white_papers/wp_mgt_antivirus_ss_smb.pdf+how+do+companies+defend+virus&hl=en&ct=clnk&cd=3&gl=us

Moir, R. (2003, October 1). Defining malware: FAQ. Retrieved August 3, 2008, from Microsoft Technet Website: http://www.microsoft.com/technet/security/alerts/info/malware.mspx

Nanoadmin. (2007, October 27). Companies are infected with malware, even though they have protection. Message posted to http://nanoscanblog.pandasecurity.com/Companies-are-infected-with-malware_2C00_-even-though-they-have-protection.aspx

Notenboom, L. A. (n.d.). What's the difference between a 'trojan horse' a 'worm' and a 'virus'?.

Business Know-How, Retrieved August 3, 2008, from http://www.businessknowhow.com/tips/virworm.htm

Pdesigns. (2006, October 11). The difference between a virus, worm and Trojan horse. Message posted to http://www.nowpublic.com/the_difference_between_a_virus_worm_and_trojan_horse

PHYSORG. (2007, March 29). Webroot study finds 43% of firms hit with malware. Retrieved August 4, 2008, from PHYSORG.com Website: http://www.physorg.com/news94386279.html

Radatti, P. V., & Wells, J. W. (2006, August, 15). *U.S. Patent No. 7,093,135* [Patent file]. Washington, DC: U.S. Patent and Trademark Office.

Roebuck, T. A. (2005, December 2). Network security: DoS vs DdoS attacks. Retrieved August 4, 2008, from Computer Crime Research Center Website: http://www.crime-research.org/articles/network-security-dos-ddos-attacks/

Schinder, T. (2004, July 23). ISA server DMZ scenarios. Retrieved August 4, 2008, from ISAServers Website: http://www.isaserver.org/tutorials/ISA_Server_DMZ_Scenarios.html

Security Focus. (2003, December 10). Worm propagation in protected networks. Retrieved June 23, 2008, from http://www.securityfocus.com/infocus/1752

Szor, P. (2005). *The art of computer virus research and defense*. Upper Saddle River, NJ: Pearson Education, Inc.

Touchstone, SC (2008). Virus samples. Retrieved August 1, 2009, from 62NDS Website: http://www.62nds.co.nz/pg/e90.php

Appendix – Graphical User Interface Development

During the development of the application, a graphical user interface will be designed for end users and administrators. The default interface has been sectioned off into five categories: internet, mail, programs, favorites, and reporting. These defaults can be modified, or even entirely removed; efficiency and usability are the major points of interest.

Table 5.1 - *Default Approved Applications*

Internet	Mail	Programs	Favorites	Reporting
Mozilla Firefox	Microsoft Outlook	Install a Program	Calendar	Ticketing
Internet Explorer	Thunderbird	Microsoft Office	Tasks	Bug Reporting
Safari			Contacts	
Opera				

Within the internet category by default, administrators have the option to allow Internet Explorer, Firefox, Safari and Opera. Within the mail category, by default Microsoft Outlook and Thunderbird are allowed. The program category is slightly different than the other categories, as it can be different for each user depending on their role within the business and how the administrators configured the application.

For example, in a company with three departments, such as Accounting, Human Resources, and Information Technology, a group of users can be assigned to the Accounting Department and thus will only be able to access and install applications approved for that department. Within this program category, the option to install an application appears as a link which takes users to the program installation page displaying a list of admin-approved applications for use within the company. After installation has completed, the user can access the installed application from the program category.

The favorites category contains a set of applications that the user has chosen from the approved programs category; the default applications for this category include Calendar, Tasks, and Contacts. Within each category, the option to select an application as a favorite is present as a yellow or blue symbol next to each application (yellow signifying a favorite and blue signifying a normal application).

The last category, reporting, includes the option to report a bug or submit a ticket to an administrator. This option can also be used to request applications to be added to the application. If the user is already an administrator they will have their own category in which they can access tickets, bug reports, requested applications, and statistics.

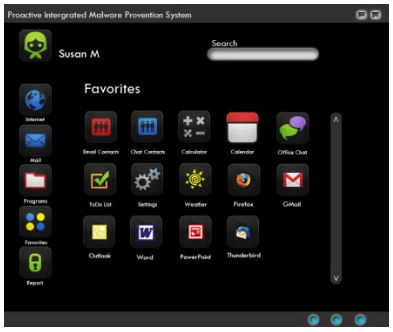

Figure 5.1 - Primary Launcher

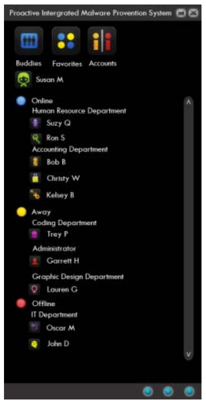

Figure 5.2 - Secondary Launcher

Administrators also have the ability to enable the chat functionality within the program, allowing users to communicate with other active users through instant messaging. The chat functionality is only available from the secondary launcher which is located within the bottom right hand corner of taskbar. The secondary launcher by default will have three different categories: buddies, favorites, and accounts. The buddy list will be sectioned off by online, offline, and away, as well as by department. The ability to enable or disable inter-office chat is a critical necessity for many businesses today. The secondary launcher also has the ability to access the user's favorite programs. The last feature is the accounts category within the secondary launcher. By default users can change their passwords, change their icons, request system notifications, or even set an alternative alias.

Graphical User Interface Screenshots

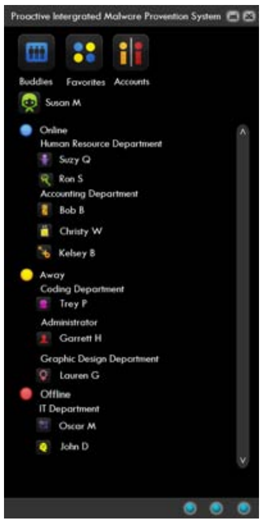

Figure 5.3 -

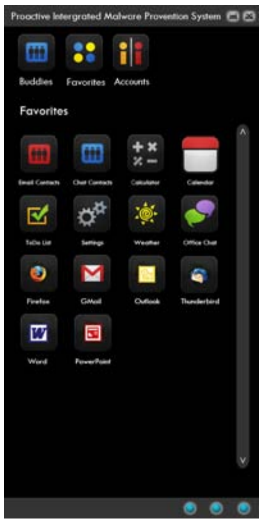

Figure 5.4 -

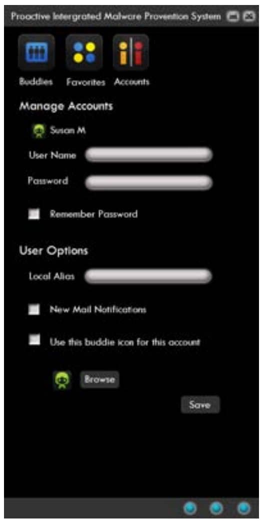

Figure 5.5 -

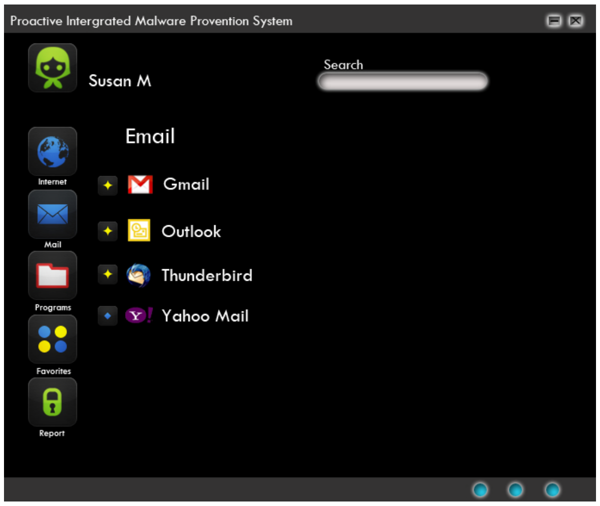

Figure 5.6 -

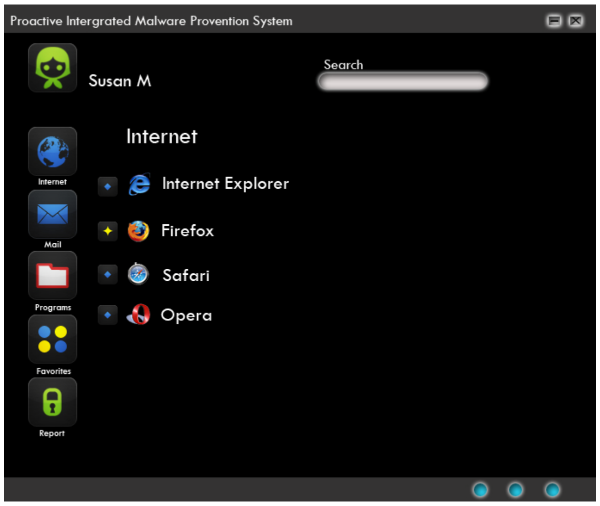

Figure 5.7 -

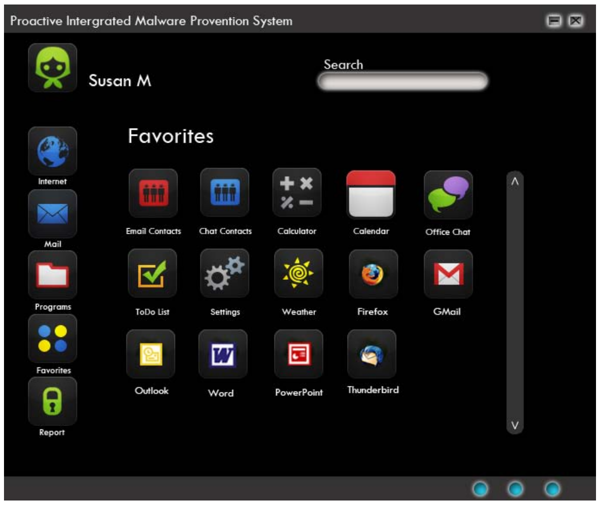

Figure 5.8 -

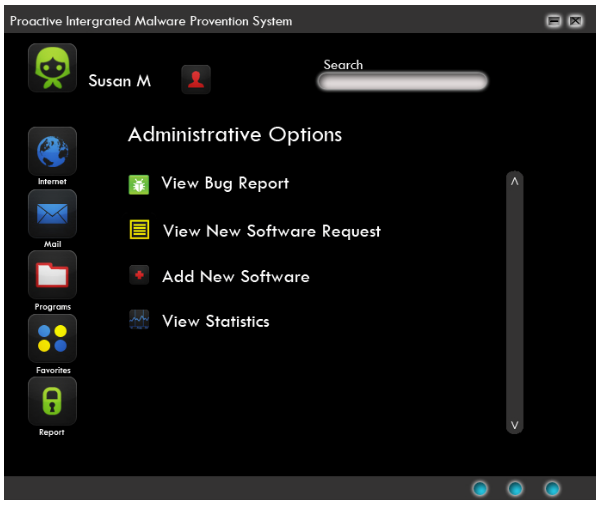

Figure 5.9 -

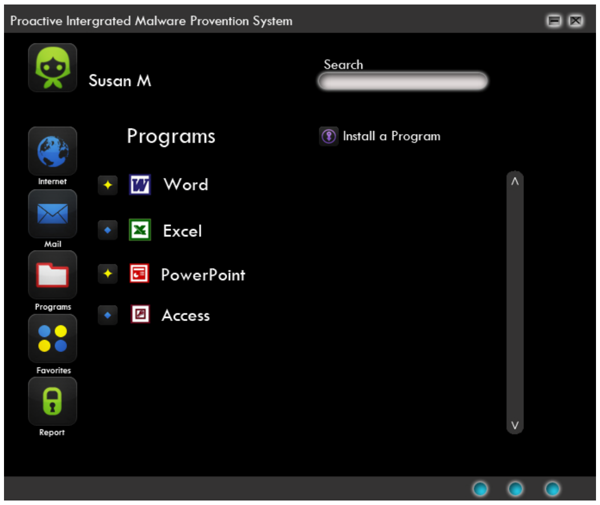

Figure 5.10 -

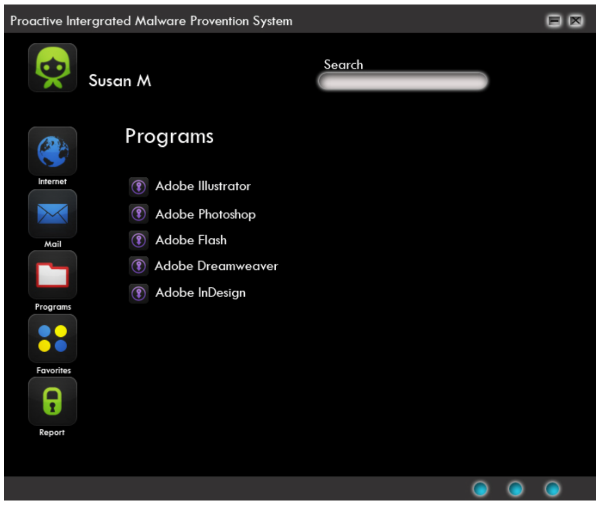

Figure 5.11 -

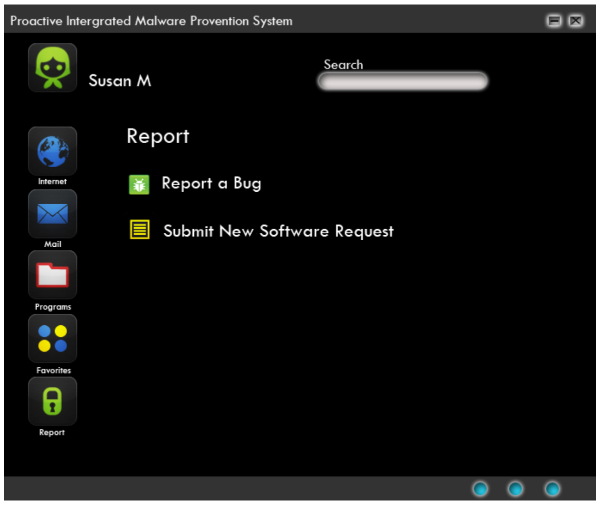
Proactive Intergrated Malware Provention System
Susan M
Search
Report
Report a Bug
Submit New Software Request
Internet
Mail
Programs
Favorites
Report

Figure 5.12 -

SIP Development Process

	Phase	Client Development	Phase	Server Development	Phase	UI Development	Phase	Document Development
Week 1 May 25 - May 29	I	Initial Phase - Research - Environment	I	Environment Setup - Network Configuration - Windows Server [Active Directory] - SQL Server Implementation Server Authentication Phase - Active Directory - SQL	I	Initial Phase - Research [Color Choice, Location, Layout] - Documentation	I	Finalization Phase - Methodology Finished - Chapters 4 & 5 Outlined & Ready
Week 2 June 1 - June 5							II	Revision 9 Phase - Faculty Review [Roy, Micah, Bill] - Review & Modify [Literature Review, Research Question, Abstract Hypothesis]
Week 3 June 8 - June 12					II	Conceptualization Phase - Concept Art - Rough Drafts	III	Revision 10 Phase - Faculty Review [Rebecca, Diane, Kelly]
Week 4 June 15 - June 19	II	Authentication Phase - List Functions - Pulling Approved List - Option to Retrieve - Request Authentication	II	Client Authentication Phase - Active Directory - SQL			IV	Revision 11 Phase - Design Improvement
Week 5 June 22 - June 26					III	Critique Phase - Instructor / Student Review - Suggestions / Modification Information Collected	V	Presentation Phase - PowerPoint Creation - Website Development
Week 6 June 29 - July 3								
Week 7 July 6 - July 10	III	Updating Phase - Schedule Updating - Server Update Checks	III	Updating Phase - Client Updates - Retrieve Client Information	IV	UI Finalization Phase - Finishing touches	VI	Polish I Phase - Review & Improve
Week 8 July 13 - July 17								
Week 9 July 20 - July 24					V	Implementation Phase - If schedule on time; then implement	VII	Polish II Phase - Review & Improve
Week 10 July 27 - July 31	IV	Testing Phase - Does it work?	IV	Testing Phase - Does it work?	VI	Testing Phase - Does it look good?	VIII	Testing Phase - Requirements met?

May 2009

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
			Week 1			
24	25	26	27	28	29	30

June 2009

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			Week 2			
31	1	2	3	4	5	Development Meeting 6
			Week 3			
7	8	9	10	11	12	13
			Week 4			
14	15	16	17	18	19	Development Meeting 20
			Week 5			
21	22	23	24	25	26	27
	Week 6					
28	29	30				

July 2009

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
			Week 6			
			1	2	3	Development Meeting 4
			Week 7			
5	6	7	8	9	10	11
			Week 8			
12	13	14	15	16	17	Development Meeting 18
			Week 9			
19	20	21	22	23	24	25
			Week 10			
26	27	28	29	30	Development Meeting 31	

2009 PROACTIVE MALWARE PREVENTION SOFTWARE