NIST Special Publication 1058

Using Host-Based Antivirus Software on Industrial Control Systems: Integration Guidance and a Test Methodology for Assessing Performance Impacts

Joe Falco, NIST
Steve Hurd, SNL
Dave Teumim, Teumim Technical, LLC.

NIST National Institute of Standards and Technology • Technology Administration • U.S. Department of Commerce

Using Host-Based Antivirus Software on Industrial Control Systems:

Integration Guidance and a Test Methodology for Assessing Performance Impacts

Version 1.0

September 18, 2006

Joe Falco, *National Institute of Standards and Technology*
Steve Hurd, *Sandia National Laboratories (SNL)*
Dave Teumim, *Teumim Technical, LLC (Consultant for Sandia National Laboratories)*

Acknowledgments

The authors wish to thank their colleagues who contributed to the development of this document and reviewed drafts. Without their insightful guidance and generous help this project would not be possible. A special note of thanks goes out to the companies that hosted site visits in support of our work, and to Pacific Northwest National Laboratory (PNNL) for providing test validation data.

The following is a partial list of companies and organizations that contributed to this project:

Aspen Technology, Inc
The Dow Chemical Company
DuPont Company
Honeywell Process Solutions
Invensys Process Systems
Invensys Wonderware
Kinder Morgan Operating L.P. "D"
McAfee, Inc
The Procter & Gamble Company
Symantec Corporation
Telvent

This work is the result of a collaborative effort between the National Institute of Standards and Technology (NIST) and Sandia National Laboratories, with funding support and guidance from the Department of Energy Office of Electricity Delivery and Energy Reliability's National SCADA Test Bed (NSTB) program.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Commercial off-the-shelf (COTS) antivirus software is used to detect and eliminate viruses and other malware on hosts, such as workstations and servers. Many end-users and vendors of industrial control systems (ICS) have concerns regarding deployment of antivirus software on these systems. The most significant concerns include:

- Antivirus software may negatively impact the time-critical control processes of an ICS.
- There is inadequate public domain information available regarding the use of antivirus on ICSs.
- There is a need for more direct contact and collaboration between commercial antivirus vendors and ICS vendors and end-users. *In the recently released DOE/DHS "Roadmap to Secure Control Systems in the Energy Sector" [1], the need for antivirus tools specifically designed and tested for ICS use is identified.*

This document is designed to help ICS end-users answer the following questions:

- What impact will antivirus software have on the performance and stability on my ICS?
- Can antivirus software be configured to minimize performance and stability impacts on my ICS?
- Should I deploy commercial antivirus software with my ICS?
- How do I deploy commercial antivirus software with my ICS?

This document is the result of a collaborative effort between the National Institute of Standards and Technology, and Sandia National Laboratories, under the guidance and sponsorship of the Department of Energy's Office of Electricity Delivery and Energy Reliability and its National SCADA Test Bed (NSTB) program. The project team collected information from ICS end-users who ranged from not using antivirus software on any of their systems to those who have developed sophisticated processes to use antivirus software on virtually all of their systems. The team also spoke with ICS vendors who ranged from distributing antivirus software with their systems to those who recommend against its use in most cases. Finally, the team engaged antivirus software vendors in addressing the use of their products on ICSs.

To help determine the impact of antivirus software on ICSs, the team designed a series of performance tests using commercially available antivirus software packages and control software. The major findings from the laboratory tests are as follows:

- Manual scanning, also known as "on-demand" scanning, has a major effect on control processes, in that they take CPU time needed by the control process (sometimes close to 100% of the CPU time). Minimizing the antivirus software throttle setting lessens, but does not remove this effect.
- Active scanning, also known as "on-access" scanning, has little or no effect on control processes.
- Signature updates can also take up to 100% of CPU time, but for a much shorter length of time than a typical manual scanning process.

These lab findings generally support the industry feedback that the team has received. Discussions of practices that control vendors and end-users currently use to contend with these issues are reflected in section 4 of this document. In many cases, performance impacts can be reduced by using configuration settings, scanning practices and maintenance scheduling that are different than those recommended for typical IT system application of antivirus software. In most cases, control vendors have specified antivirus software configuration settings for use with their lines of products.

This document provides the industrial controls community with:

- A collection of background information on ICSs and antivirus software for IT and control system professionals who are responsible for securing these systems.
- Implementation guidance and "good practices" with a focus on minimizing performance impacts. *Guidance is based primarily on knowledge gathered from ICS end-users and vendors who are using antivirus software as a security component in their ICSs and have already wrestled with the many performance implications associated with its use.*
- A methodology for developing custom performance test procedures for assessing ICSs for any performance impacts associated with antivirus software practices.

This work has assembled ICS based antivirus knowledge into a single document and serves as a starting point or as a secondary resource when installing, configuring, running, and maintaining antivirus software on an ICS. This collaborative industry effort has also made antivirus software vendors more aware of ICSs and their special performance requirements, initiating better communications between the two fields.

The DOE "Roadmap to Secure Control Systems in the Energy Sector" includes two priorities pertaining to the use of antivirus software with ICSs. The first is to make available and disseminate field-proven best practices for control system security, which this document will help fulfill. The second is to develop cost effective antivirus protection that minimizes host impact. The working relationship established with the antivirus vendors participating in this project creates an environment for this future collaboration.

Note: This document focuses exclusively on host-based, antivirus software use in ICSs. This document does not focus on the use of other important security technologies, such as gateway antivirus tools, firewalls or intrusion detection systems. The authors want to stress this should not be interpreted as a recommendation to use antivirus software without careful evaluation. Further, we stress this should not be interpreted as a recommendation to use antivirus software in place of other security technologies. Ultimately, a combination of security technologies in a layered defense strategy is typically the best solution for securing ICSs.

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

End-users and vendors of industrial control systems (ICSs) have expressed concerns that the deployment of antivirus software may interfere with the operation of time-critical control processes. These concerns are one of the reasons that antivirus software has not been more widely adopted in these industries. This document is intended to help to minimize as well as measure the performance impacts caused by the addition of antivirus software on ICS servers and workstations. This document does not evaluate the performance of antivirus software products, nor does it imply the use of any particular brand of antivirus software[1]. This work is the result of a collaborative effort between the National Institute of Standards and Technology (NIST), Sandia National Laboratories, under the guidance of the Department of Energy Office of Electricity Delivery and Energy Reliability's National SCADA Test Bed (NSTB) program [2].

The project team collected information from end-users who range from not using antivirus software on any of their systems to those who have developed sophisticated processes to use antivirus software on virtually all of their systems. The team also spoke with ICS vendors who ranged from distributing antivirus software with their systems to those who recommend against its use in most cases. Finally, we engaged antivirus software vendors in addressing the use of their products on ICSs. Information collected includes:

- Control system component configurations and network architectures
- ICS performance requirements for several industry sector applications
- Current industry practices and problems using antivirus with ICS components

The information collected from this process along with a study conducted within the NIST Industrial Control Security Test Bed [3] is the basis for the development of these guidelines and testing methodology.

To determine the impacts of antivirus software on ICSs, NIST conducted a series of performance tests using commercially available antivirus software, control system software, and hardware within its Industrial Control Security Test Bed. The combined results of this work were used to produce a set of guidelines and a test methodology for industry. The guidelines, focused on minimizing performance impacts, are based on the expertise of ICS end-users and vendors who are using antivirus software as a security component in their ICSs as well as the developers of antivirus software. The test methodology provides a general set of procedures for use by industry as a starting point when developing control system specific performance impact tests. A laboratory test bed was used to demonstrate the impact of antivirus software and obtain performance data in support of this effort.

[1] Commercial equipment and materials are identified in order to adequately specify certain systems. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, Sandia National Laboratories, or the Department of Energy, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

This document combines many aspects of ICS based antivirus knowledge into a single document and serves as a starting point or as a secondary resource when installing, configuring, running, and maintaining antivirus software on an ICS. The document contains:

- A collection of background information on ICSs and antivirus software for both control system and IT professionals who are responsible for securing ICSs.
- Implementation guidance and "good practices" with a focus on minimizing performance impacts. *Guidance is based primarily on knowledge gathered from ICS end-users and vendors who are using antivirus software as a security component in their ICSs and have already wrestled with the many performance implications associated with its use.*
- A test methodology and example test cases to develop custom performance test procedures for assessing ICS for performance impacts associated with antivirus software practices.

1.2 AUDIENCE AND ASSUMPTIONS

This document is primarily intended to help individuals responsible for installing, configuring, and maintaining antivirus products on control system workstations and servers within the following company perspectives:

- End-users who have never implemented antivirus on their systems for fear that it may disrupt their production
- End-users implementing a different antivirus application than that specified and certified by the control system vendor
- End-users concerned with the effects that antivirus software may have on the performance of their control systems
- Control system vendors implementing, specifying, or certifying the use of antivirus software on their systems

1.3 DOCUMENT ORGANIZATION

This document is divided into six sections followed by five appendices. The following is a listing describing the document structure.

Section 1: purpose, audience, and document structure.
Section 2: overview of industrial control systems.
Section 3: functional overview of commercial antivirus software.
Section 4: guidance for deploying antivirus software.
Section 5: test methodology for ICS performance impacts testing.
Section 6: efforts underway to continue this work.
Appendix A: example test cases demonstrating use of the test methodology.
Appendix B: listing of performance software.
Appendix C: listing of acronyms and abbreviations.
Appendix D: glossary of terms.
Appendix E: references.

1.4 SUMMARY OF PROJECT RESULTS

There were two efforts used to develop this document. The first was an information gathering process involving ICS end-users and vendors who are actively engaged in recommending and using antivirus software on their systems. Antivirus software vendors were also involved in these activities. The second was a laboratory effort to establish a set of generic test procedures for evaluating the performance impacts that antivirus software may have on an ICS.

1.4.1 INFORMATION COLLECTED FROM INDUSTRY

Several ICS vendors and end-users were interviewed by telephone and site visits. Table 1 summarizes the main points of the information collected, contrasting the ICS industry standing with regard to the use antivirus in 2005 to that in 2003.

Use of antivirus software: attitudes in 2003	Use of antivirus software: status in 2005
Widespread fear that antivirus software will cause an ICS to fail	The trend is that end-users are increasingly installing antivirus software on new ICSs (and to varying degrees on legacy ICSs). However, many end-users still decide not to use antivirus software on their systems.
Minimal support from ICS vendors on the use of antivirus software with their products	The degree of support varies with vendor: ▪ Supplied, configured and tested with ICS ▪ Certified with use instructions ▪ Good to use but not certified ▪ Use at your own risk In general, support is increasing over time.
Antivirus software will use critical computing capacity required by the ICS to meet its performance requirements	Despite increased use, there are gaps in performance testing. (The test methodology included in this document is intended to help fill this gap.)
Antivirus software updates and maintenance are too much trouble and are time consuming	Antivirus updates and maintenance practices are still issues due to the continuous operation requirements of most ICSs.

Table 1: Use of Antivirus Software on ICSs

1.4.2 TEST METHODOLOGY RESULTS

The test methodology was based on information gathered from industry and research performed on the NIST Industrial Control Security Test Bed. The methodology is made up of five sets of procedures to assess control system platform performance using a typical antivirus software package on both workstations and servers hosting ICS software. The test methodology should be used as a starting point when developing control system specific performance impact tests.

The test methodology was validated on two government laboratory test beds. Validation was performed on the NIST test bed on ICS components typically found in the process controls industry and on a

SCADA test bed located at DOE's Pacific Northwest National Laboratory (PNNL). Results gathered from this validation testing were compiled into a collection of test cases that demonstrate use of the methodology and provide example data. Observation of the data from the first two test cases is as follows:

- Manual scanning, also known as "on-demand" scanning, has a major effect on control processes, in that they take CPU time needed by the control process (sometimes close to 100% of the CPU time). Minimizing the antivirus software throttle setting lessens, but does not remove this effect.
- Active scanning, also known as "on-access" scanning, has little or no effect on control processes.
- Signature updates can also take up to 100% of CPU time, but for a much shorter length of time than a typical manual scanning process.

2 OVERVIEW OF INDUSTRIAL CONTROL SYSTEMS

2.1 SYSTEM DESCRIPTIONS

Industrial Control Systems (ICS) is a general term that encompasses several types of control systems, including supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and other smaller control system configurations such as skid-mounted Programmable Logic Controllers (PLC) often found in the industrial sectors and critical infrastructures. [4] [5]. ICSs are used in the electric, water, oil and gas, chemical, pharmaceutical, pulp and paper, food and beverage, and discrete manufacturing (automotive, aerospace and durable goods) industries. The material presented in this section provides a general description of SCADA and DCS, their performance requirements and some security aspects pertaining to malware protection.

2.1.1 SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA)

SCADA systems are highly distributed systems used to control geographically dispersed assets, often scattered over thousands of square kilometers, where centralized data acquisition and control are critical to system operation. They are used in the distribution operations of water supply systems, oil and gas pipelines, electrical power grids, and railway transportation systems. A SCADA control center performs centralized monitoring and control for field sites over long distance communications networks. This includes monitoring alarms and processing status data. Based on information received from remote stations, automated or operator-driven supervisory commands can be pushed to remote station control devices, which are often referred to as field devices. Field devices control local operations such as opening and closing valves and relays, collecting data from sensor systems, and monitoring the local environment for alarm conditions.

A simplified example of a SCADA control center connected with two field stations is shown in Figure 1. A control center sends commands and receives status data over a long distance communications network. The key control components within the control center are the SCADA server, a Human Machine Interface (HMI), and a data historian. The SCADA server issues commands and requests for data to field devices, and interfaces with the HMI for operator monitoring and control capability. Operators at the Control Center HMI monitor displays of data relayed from the field, react to alarm conditions, and initiate high-level commands such as new set points to field sites. In some cases, there may be HMIs local to the field site. The data historian is the central repository for SCADA system data. Some SCADA servers orchestrate a network containing hundreds of field sites.

2.1.2 DISTRIBUTED CONTROL SYSTEMS (DCS)

DCSs are used to control production systems within the same geographic location for businesses such as oil refineries, gas processing plants, electric power generation plants, chemical plants, automobile production facilities, and food and pharmaceutical processing facilities. These systems may be applied to control continuous, batch or discrete manufacturing processes. A DCS uses a centralized supervisory control loop to mediate a group of localized controllers that share the overall tasks of carrying out an entire production process. In addition to supervisory level and field level control, intermediate levels of control may also exist.

An example of the components and network architecture for a typical DCS is shown in Figure 2. The key control components at the supervisory level are a control server, a Human Machine Interface (HMI),

and a data historian. The control server communicates with its subordinates via a control network. Like the SCADA control server, the DCS supervisor sends set points to and requests data from the distributed

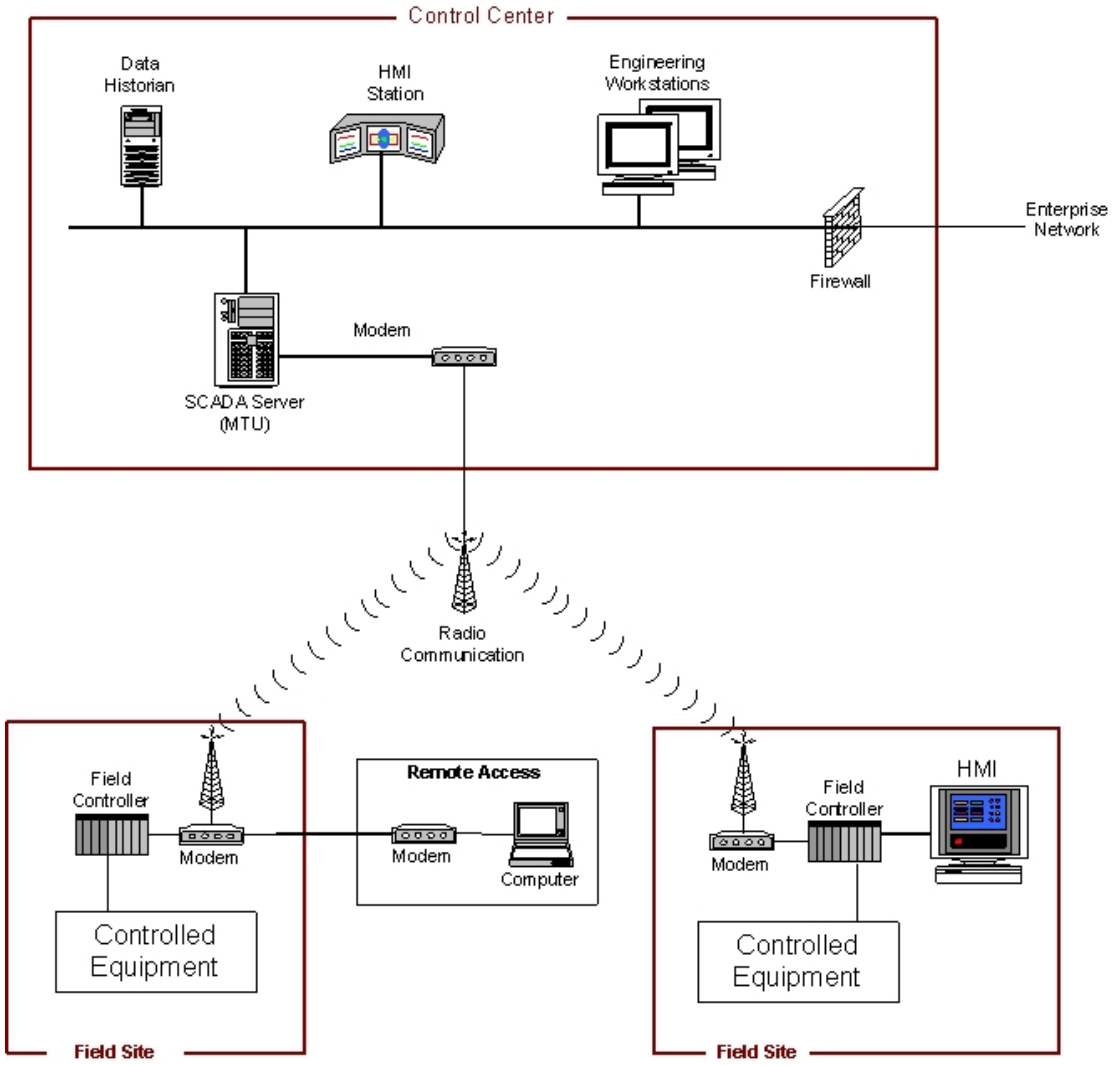


Figure 1: SCADA System Example

field controllers. The distributed controllers control their process actuators based on control server commands and sensor feedback from process sensors. Operators at both the supervisory and local field level HMI monitor data, react to alarm conditions, and initiate high level commands such as new set points to field sites. The data historian is the central repository for factory or plant data.

2.1.3 PERFORMANCE REQUIREMENTS

Control systems used in distribution and manufacturing industries use the same type of components and are similar in operation. One of the primary differences is the fact that DCS-controlled sub-systems, when compared to geographically dispersed SCADA field sites, are usually located within a more confined factory or plant centric area. Communications are usually performed using local area network (LAN) technologies that are typically more reliable compared to the long distance communication systems used by SCADA systems. DCS systems usually employ a greater degree of closed loop control

than SCADA systems since control of manufacturing processes is typically more complicated than the control of a remote distribution process.

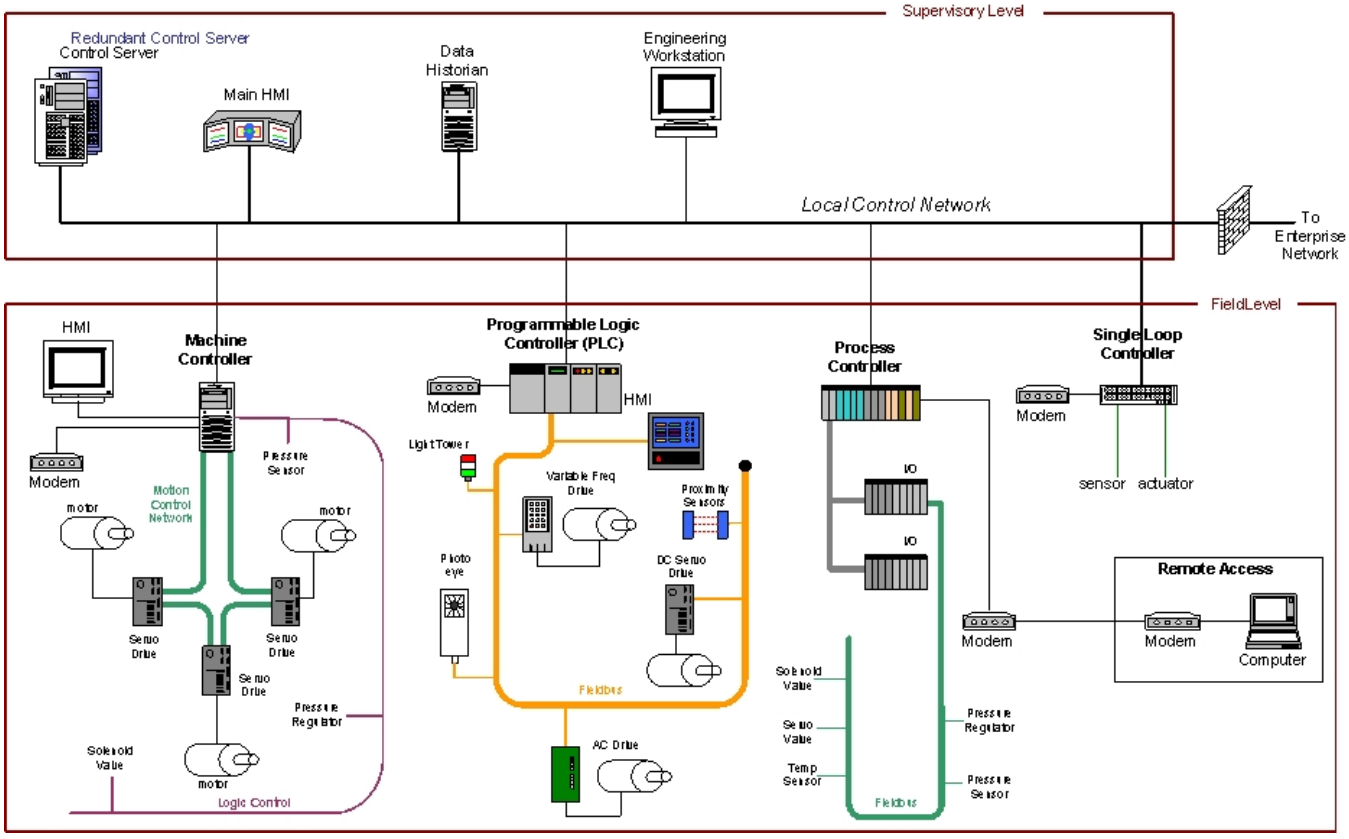


Figure 2: DCS Example

SCADA, DCS and other ICS configurations may contain a proliferation of control loops structured in a layered network architecture. Supervisory-level loops and lower-level loops operate continuously over the duration of a process with cycle times ranging on the order of minutes to milliseconds. These various degrees of real-time control are often intolerable to delay. In order to maintain correct ICS operations, care must be taken to insure that critical data is delivered so that the process is maintained and safety mechanisms are given the appropriate computing resources.

2.2 IT SECURITY

2.2.1 VULNERABILITIES

Now that ICSs are being interconnected with business systems and are being serviced using remote Internet access capabilities, they are now vulnerable to remote cyber attack. In many production and distribution systems, the ICS is interfaced with the enterprise level of the organization in order to provide business operations a view of the operational environment.

Most ICSs in use today were developed years ago, before public and private networks, desktop computing, and the Internet started to be used as a common part of business operations. These older ICSs were designed to meet performance, reliability, and safety requirements of their day and in many

cases were physically isolated and based on proprietary hardware, software, and communication protocols that were not well understood outside the ICS community. The need for cyber security measures within these systems was not anticipated, and when addressed, security for ICSs meant physically securing access to the network and the consoles that controlled the systems and performing security background checks for all personnel able to touch the systems.

As microprocessor, personal computer, and networking technology evolved during the 1980's and 1990's, the design of ICSs changed to incorporate the latest technologies. Internet-based technologies started to be integrated into ICS designs in the late 1990's. Also, COTS hardware, software, and networking capabilities were being used in ICSs, with the trend toward using COTS further down the control network architecture, approaching and including the final control device (sensor or actuator). This trend opens ICSs to anonymous access by persons without background checks who are located outside the physical security controls of the site. This leads to exploitation through software and networking vulnerabilities that are published on the web in security and hacker sites. While these increased vulnerabilities are leading to the integration of new and existing security technologies into ICSs, special care must be taken to ensure that the addition of these technologies does not affect ICS performance requirements.

2.2.2 THE THREAT OF MALWARE

The term malware covers a wide range of malicious software designed to take over and/or damage a computer's operating system, applications or data [6]. Malware spreads to computer systems via removable media and network connectivity. Once malware is present on the organization's network, it can easily spread from computer to computer.

When a system becomes infected with malware, it can be very difficult to detect, remove, and repair damage. Depending on the severity of the type of infection, the damage can range from a minor annoyance (e.g., slower responsiveness) to loss of integrity (e.g., alteration of data, alteration of application functionality) to irreparable damage (e.g., loss of data and service).

Malware is mainly, but not exclusively, composed of viruses, worms, Trojan horses, malicious mobile code, and blended attacks. Malware also includes hacker tools such as backdoors, rootkits, and keystroke loggers, and tracking cookies used as spyware. As a quick introduction to malware, descriptions of viruses, worms, Trojan horses, malicious mobile code, and blended attacks are given below.

VIRUSES are typically spread through the process of file or program sharing by hiding inside content being communicated between persons (e.g., a legitimate-looking email attachment). A virus spreads from computer to computer by inserting itself into a host program or data file; it then self-replicates—makes copies of itself—and distributes them to other files, programs, or computers. There are many different forms of viruses – for example, a boot sector virus hides in the boot sector of a hard drive or removable media, while a file infector virus usually appends itself to a legitimate program, such as a word processor. A macro virus attaches itself to an application document, such as a word processing file or a spreadsheet, and uses that application's macro programming capability (e.g., Microsoft Office applications use Visual Basic for Applications [VBA]). When a virus is triggered, such as executing an infected program or opening an infected document, the virus delivers its payload. This payload usually includes a replication process as well as actions that may cause damage to the operation of a computer.

Damage caused by a virus can include deletion or intentional corruption of files, corruption of system areas to prevent rebooting, and degradation of computer performance by theft of memory, disk space, or clock cycles, as well as through system modifications.

WORMS are self-replicating, self-propagating, self-contained malicious programs that automatically spread from one computer to another through a network. Unlike a virus, a worm does not need to insert itself into a program or file to replicate. Although some worms are designed to spread by the human-initiated execution of "worm code", most worms can execute and spread without user intervention. Worms may also deliver a malicious payload in addition to replicating. As the process of self-replication increases, memory and networks become overwhelmed, possibly causing systems or networks to stop functioning, an event called a denial of service attack.

TROJAN HORSES are non-replicating programs that appear to be benign but actually have a hidden malicious purpose. When these programs are run, they use the authorizations of the executing user to cause damage or allow unauthorized access to the local system. Some Trojan horses are intended to replace existing programs, while others present themselves as new programs. The functions performed by Trojan horses differ; some perform both the benign function they are supposed to perform and a malicious function, while others perform only malicious actions. For instance, a user could download and run a screen saver utility that actually deletes the computer's files. Another example of severe damage caused by Trojan horses is creating a backdoor that gives malicious users access to the infected computer system.

MALICIOUS MOBILE CODE is malicious software that is transmitted from a remote system to a local system, typically without the user's explicit instruction. Popular languages for malicious mobile code include Java, ActiveX, JavaScript, and VBScript. Programs written using mobile code can be used by many different operating systems and applications, such as Web browsers and e-mail clients. Malicious mobile code is increasingly used to attack systems, as well as to transmit viruses, worms, and Trojan horses to users' systems. Malicious mobile code differs significantly from viruses and worms in that it does not infect files or attempt to propagate itself. Instead of exploiting particular vulnerabilities on a system, it often affects systems by taking advantage of the default privileges granted to mobile code.

A BLENDED ATTACK uses multiple infection or transmission methods, such as combining virus and worm characteristics. For example, a worm may use self-replication to rapidly distribute a virus that will infect systems upon execution. This combination of techniques often allows blended attacks to spread more rapidly and cause more widespread damage than threats that use only a single infection or transmission method.

Although each of the above malware threats are different, and each tends to be defined by the way it propagates, the antivirus software industry often groups these three types of malware under the more common name "viruses", and the software that detects, responds and removes this called "antivirus software". This convention of grouping everything together under "virus" will be used throughout this document unless otherwise indicated. It should also be mentioned that there are also other programs that fight malware, such as antispyware software, rootkit detectors, and specialty utilities designed to detect and eradicate only a single type of malware (e.g., a particular widespread worm). Similar to antivirus software, these products may also cause performance impacts on systems.

2.2.3 EXAMPLE MALWARE ATTACKS ON ICSS

Below are published accounts of malware incidents that affected industrial control systems:

CSX Train Signaling System – Virus Infection[2]: In August of 2003, the Sobig computer virus was blamed for shutting down train signaling systems throughout the east coast of the U.S. The virus infected the computer system at CSX Corp.'s Jacksonville, Florida headquarters, shutting down signaling, dispatching, and other systems. According to an Amtrak spokesman, ten Amtrak trains were affected in the morning. Trains between Pittsburgh, Pennsylvania and Florence, South Carolina were halted because of dark signals, and one regional Amtrak train from Richmond, Virginia to Washington DC and New York City was delayed for more than two hours. Long-distance trains were delayed between four and six hours.

Manufacturing Plants – Worm Infection[3]: In August 2005, the Zotob and Pnp worms knocked 13 of DaimlerChrysler's U.S. automobile manufacturing plants offline for almost an hour, idling workers as infected Microsoft Windows® systems were patched. Plants in Illinois, Indiana, Wisconsin, Ohio, Delaware, and Michigan were also knocked offline. While the worm affected primarily Windows 2000, it also affected some early versions of Microsoft XP. Symptoms included the repeated shutdown and rebooting of a computer. Zotob and its variations also caused computer outages at heavy-equipment maker Caterpillar Inc., aircraft-maker Boeing, and several large U.S. news organizations.

Nuclear Power Plant – Worm Infection[4]: In August 2003, the Nuclear Regulatory Commission confirmed that in January 2003, the Microsoft SQL Server worm known as Slammer infected a private computer network at the idled Davis-Besse nuclear power plant in Oak Harbor, Ohio, disabling a safety monitoring system for nearly five hours. In addition, the plant's process computer failed, and it took about six hours for it to become available again. Slammer reportedly also affected communications on the control networks of at least five other utilities by propagating so quickly that control system traffic was blocked.

2.2.4 DEFENSE-IN-DEPTH

A single security product or technology cannot adequately protect an ICS, so a multiple layer strategy involving two (or more) different overlapping security mechanisms is desired to avoid a vulnerability in one technique from allowing a compromise. Securing an ICS is based on a combination of effective security policies and a properly configured set of technical security controls [5]. An effective cyber security strategy for an ICS should apply this layered strategy, which is known as defense-in-depth. This strategy uses security mechanisms that are layered so that the impact of a failure in any one mechanism is minimized.

---

[2] Additional information on the CSX Train Signaling System incident can found at: http://www.cbsnews.com/stories/2003/08/21/tech/main569418.shtml and http://www.informationweek.com/story/showArticle.jhtml?articleID=13100807

[3] Additional information on the Zotob Worm incident can found at: http://www.eweek.com/article2/0,1895,1849914,00.asp and http://www.computerwire.com/industries/research/?pid=750E3094-C77B-4E85-AA27-2C1D26D919C7

[4] Additional information on the Davis-Besse incident can found at: http://www.taborcommunications.com/hpcwire/hpcwireWWW/03/0905/105866.html and http://www.securityfocus.com/news/6767

In a typical ICS this means a defense-in-depth strategy that includes:

- Developing security policies, procedures, and educational material that apply specifically to the ICS.
- Addressing security throughout the lifecycle of the ICS from architecture to procurement to installation to maintenance to decommissioning.
- Implementing a network topology for the ICS that has multiple layers, with the most critical communications occurring in the most secure and reliable layer.
- Providing logical separation between the corporate and ICS networks (e.g., stateful inspection firewall(s) between the two networks).
- Employing a DMZ network architecture (i.e., prevent direct traffic between the corporate and ICS networks).
- Ensuring that critical components are redundant and are on redundant networks.
- Designing critical systems for graceful degradation (fault tolerant) to prevent catastrophic cascading events. In addition, design systems to fail securely.
- Disabling unused ports and services on ICS devices after testing to assure this will not impact ICS operation.
- Restricting physical access to the ICS network and devices.
- Restricting ICS user privileges to only those that are required to perform each person's job (i.e., establishing role-based access control and configuring each role based on the principle of least privilege).
- Considering the use of separate authentication mechanisms and credentials for users of the ICS network and the corporate network (i.e., ICS network accounts do not use corporate network user accounts).
- Using modern technology, such as smart cards for Personal Identity Verification (PIV).
- Implementing security controls such as antivirus software and file integrity checking software, where technically feasible, to prevent, deter, detect, and mitigate the introduction, exposure, and propagation of malicious software to, within, and from the ICS.
- Applying security techniques such as encryption to ICS data storage and communications.
- In as expeditious a manner as possible, deploy security patches after testing all patches under field conditions on a test system if possible, before installation on the ICS.
- Tracking and monitoring audit trails on critical areas of the ICS.

A defense-in-depth approach would include the deployment of antivirus software on workstations, servers, and networks[7]. Antivirus software must be kept running full-time and only functions effectively when installed, configured, and maintained properly against the state of known malware attack methods and payloads. While antivirus tools are a common security practice in IT computer systems, their use with an ICS may require adopting special change management practices as well as compatibility and performance impact checks.

3 OVERVIEW OF ANTIVIRUS SOFTWARE

3.1 ANTIVIRUS SOFTWARE

Antivirus products evaluate files on a computer's storage devices against an inventory of known virus signature files[8][9]. If one of the files on a computer matches the profile of a known virus, the virus is removed through a disinfection process so it cannot infect other local files or communicate across a network to infect other files. Antivirus software can be deployed on workstations, servers, firewalls and handheld devices and may be deployed in three modes:

- **Workstation Installation –** The software is installed and running on a workstation to protect it against network or detachable media.
- **Server Installation –** The software is installed and running on a shared server to monitor data passing through the server or for viruses resident on servers that can rapidly spread to supported workstations.
- **Boundary Installation –** The software is installed at the logical or physical boundaries of a network or system such as on a firewall.

3.1.1 SCAN ENGINE

The scan engine is the "active workhorse" of the antivirus software package. It scans the hard disk, memory, and removable storage devices of a host computer to detect and report viruses. The engine scans components based on built-in detection rules, for instance, when a file is opened (read access) or closed (write access). Upon startup an engine will scan the boot sectors of the hard disk, as well as memory. A scan engine may only scan files with certain types of file extensions, for instance "exe" or "com" files, based on a user selected configuration and the assumption that these files are the most susceptible to being exploited by malware.

A scan engine may need to do extensive preprocessing of a file to make it ready for a scan. For instance, the code to be scanned may be embedded in a zip file, and the file will have to be unzipped before scanning. There may be many levels of processing involved (for instance a zip file containing a smaller zip file).

3.1.2 SCANNING TECHNIQUES

In general, commercial antivirus software packages perform three types of scanning: signature matching, heuristic analysis, and behavior blocking. Each vendor has a unique implementation of each technique, many of which are patented. A description of each scanning technique is given below.

- **Signature matching** is a technique that matches file code against a frequently updated database containing thousands of signatures of known viruses. Antivirus vendors continuously collect malware specimens from which they derive a signature (fingerprint) of the virus. When an antivirus scan engine looks at a file to see if it contains malware, it compares a small, condensed snippet of code determined by the antivirus vendor to characterize the virus code against the whole scanned file. The scanning process does not necessarily have to look at all the code in a file, for instance many viruses leave the original file contents intact and append the malware code at the end of the file. To be effective, the antivirus vendor must publish new virus signature files as soon as a new virus is discovered, and these new virus signature files must be downloaded immediately to all the computers that need to be protected. As a result, databases are frequently

updated and distributed to systems running antivirus software in order to keep up to date with newly released malware.

- **Heuristic analysis** is a technique designed to overcome the limitations associated with the timely availability of virus signatures for signature-based detection. Heuristics look at the anticipated behavior of computer code for unexpected actions. Examples of unexpected actions include attempts to access the boot sector, locate all documents in a current directory, write to an EXE file and delete hard drive contents. Based on these unexpected actions, a weight is assigned to each file analyzed. If the weight exceeds a certain threshold, the scanner identifies the file as containing malicious code. Heuristic analysis techniques are used to find unknown viruses that have not yet been cataloged with signatures and viruses that are designed to automatically change during propagation. Heuristics can use a significant amount of computer resources and are prone to false positives.

- **Behavior blocking** is a technique designed to look for attack behaviors, such as attempts to open a network port, in contrast to heuristics, a method that analyses code behavior.

Antivirus applications can be configured to use a combination of these scan techniques with user selectable options including scan methods to use, file types to scan and scan sensitivity levels.

3.1.3 SCAN ENGINE PERFORMANCE

Two of the major antivirus engine performance metrics involve percent coverage and scan time. Both are crucial in the design of scanning algorithms. Coverage is the percent of known virus variants the engines will detect. Viruses are categorized into two types:

- **Zoo viruses** are viruses carefully contained in libraries maintained by antivirus researchers that exist for testing and have never infected a real world computer system

- The term **in-the-wild virus** refers to a virus that has been reported by at least two separate reporting agencies to Wildlist.org. The "Wildlist" is comprised of antivirus experts throughout the industry to maintain a *snapshot* of current infections. Published monthly, the list is also used as a benchmark by some antivirus certification agencies

Antivirus software is expected to successfully detect a very high percentage of both zoo and in-the-wild viruses, usually greater than 99%. Organizations such as Virus Bulletin (VB) regularly test antivirus software packages against reference collections of both zoo and in-the-wild viruses published.

An important property of a scanning algorithm is its ability to process and scan a file quickly, so as to not hinder the opening of an executable or other file when needed by the computer user. For example, when a spreadsheet or application program is needed, the files associated with the application ("exe", "com", "dll", and data files) must be scanned and cleared as safe to use prior to the application startup and loading of the data file. The time associated with this process, usually in the order of milliseconds, is the time a user or process must wait until application startup begins. In control system use, time to scan becomes critical since all processes are contending for CPU usage and some production processes may be time critical. Therefore, the preprocessing and signature comparison process must be configured for fast, efficient operation to minimize performance impacts. An important consideration is that the file scanning required by an antivirus process may impact the performance of an ICS process.

In addition to zoo and in-the-wild viruses, the EICAR test virus [10] is a non-viral test file that antivirus software will react to as if it were a virus. The EICAR test virus provides a standardized test file for signature based virus detection software. This file can be used to verify the correct operation of antivirus software. The file is a legitimate DOS program, and prints "EICAR-STANDARD-ANTIVIRUS-TEST-FILE!" when run. The EICAR test virus is available as a text file and two archived zip files to test for viruses in multiple-level archives.

3.1.4 SCANNING MODES

Antivirus software packages typically have two available modes to perform scans. These modes, called active and manual scanning throughout this document, are referred to with different naming conventions across the different antivirus applications. The operation of these two scan modes is described as follows.

- **Active scanning** examines files, removable media, and system areas when information is accessed or changed. These actions or changes include copying and saving files, and starting applications by opening .EXE or .COM files and their associated Dynamic Link Libraries (DLL) and data files. If actions or changes are detected, the associated object is scanned for viruses using signature matching and heuristic analysis techniques. Active scanning is a serial process, preventing any application to start or file to be used prior to completion of the scanning process on it.

- **Manual scanning** detects viruses using signature matching in selected groups of files and/or directories and may be initiated by a user or automatic scheduler. For instance, a directory, group of directories, or a whole CD or hard disk may be scanned. Manual scanning generally takes longer than active scanning since a large number of files are usually scheduled for scanning. This type of scanning is used to perform "deep" scans at scheduled intervals. Configuration controls (often termed throttling) are often included to allow a user to increase the amount of CPU time to be dedicated to this process in order to complete a scan faster, or decrease the amount of system resources used in order to minimize the impact on the user and other processes.

Active scanning operates continuously and does not require intervention by a user. Since it addresses individual files, this method tends to consume significantly less resources than the manual scanning process that typically involves batch scanning on volumes of files. Both scanning modes use the same definitions database.

3.2 REMOVAL TECHNIQUES

Once an antivirus application detects a virus, there are several methods available for remediation.

- An **immediate clean and restore** technique removes virus code and restores the infected file to original condition for immediate use. However, some viruses modify registry settings, have intricately woven malware code with good code, or have permanently deleted original code. In these cases, files cannot be immediately restored and the original file may have to be reconstructed by remounting original software or restoring it from backup.
- A **delete** technique completely removes an infected file from a computer system.
- A **quarantine** technique moves infected files to a safe area on the computer where they cannot infect other files. The files may be re-introduced back into the system should an administrator determine they do not contain malware.

- A **detect and notify only** technique leaves a virus in place. The software notifies the user of the infection.

3.3 CONFIGURATION

Most antivirus applications are configurable by a client user or by a centralized server. Several of the features that are configurable within an antivirus software application are listed below:

- Timing and degree of active or manual scanning
- Location of virus definition download server so customer can stage their own vetted signature sets
- Method and timing of scan engine and signature file updates
- Enable and set the degree of heuristic analysis
- Selection of file extensions to be scanned
- Selection of file extensions to be excluded from scan
- Adjustment of CPU time dedicated to the antivirus application during manual scanning operations, sometimes called "throttling"
- Selection of methods to handle files once a virus is detected (see Section 3.2, Removal Techniques)
- Degree of client autonomy (permissions for client user to adjust configuration settings)

3.4 MAINTENANCE

Maintenance is an important aspect of antivirus software. Since new viruses are continuously found in-the-wild with some spreading around the globe in minutes or hours, it is necessary when implementing antivirus software to carefully plan how engines and signatures will be updated. Antivirus software signature files must be updated frequently in order to have the capability to detect all known viruses. Antivirus vendors typically update their signature files on their corporate web or File Transfer Protocol (FTP) sites as frequently as hours (with a large, fast spreading outbreak), or otherwise daily or even weekly. Antivirus engines are updated on a slower schedule in the time frame of months.

3.4.1 RESPONSE TO OUTBREAKS

Antivirus software companies rely on a cooperative voluntary incident reporting mechanism through an industry association called The Wildlist Organization International (Wildlist.org) for information on new virus outbreaks, also called in-the-wild viruses. Using a sample of the infection code obtained from Wildlist, antivirus software developers analyze the code in a lab, identify portions of code peculiar to that virus, and develop a compact fingerprint (signature) of the virus.

The speed with which antivirus vendors develop and deploy new signatures to their end-users may vary widely, from a few hours to a few days. These variations are due to the fact that the vendors must acquire the malware, analyze it, develop signatures, test the signatures, test the product's behavior with the signature update, then make the update available to customers. Thus the elapsed time when the virus is first detected to the time when the signature file is updated for end user downloads varies, and determines whether a fast spreading virus will be caught in time by end-users, or be applied after many corporate machines are already infected. It has been reported that response delays of just a few hours can have significant impact on the amount of damage a virus outbreak can cause in an organization due to the often exponential growth of infected messages or machines found in major virus outbreaks. In a

comparative review of antivirus response time published in February 2004 Virus Bulletin, the article quotes statistics for the Sober.C worm, discovered in Germany in December 2003. The first signature was out for one vendor's customer update site within 10 hours, 20 minutes of discovery. However many antivirus companies didn't have updated signatures available until one to two days later.

Although antivirus companies strive to respond to new virus outbreaks quickly, experienced hackers are becoming faster at exploiting vulnerabilities and are sometimes even the first to discover one. Often, exploits are released before a vulnerability is publicly identified, a "zero-day" exploit. These issues are making the use of heuristic analysis techniques and other protections (generalized as "zero-day" protections) ever more important to detect new viruses.

3.4.2 SIGNATURE UPDATES

Antivirus companies release signature file updates (also termed: virus definition updates) daily and sometimes release them multiple times in a single day, if warranted by new virus outbreaks. Signature files may be downloaded based on a full signature database update or using incremental updates based on previous updates. Full updates are large and take significantly more time installing than incremental updates; however, they are usually required if incremental updates are not performed on a regular basis. Note that incremental updates can also be significant in size. Occasionally, a new scan engine is released. Large corporate IT groups often set up a Web or FTP download script to download new signature files as well as run some screening tests with typical corporate IT applications on various platforms. This testing is used to determine if a new signature file or engine causes malfunctions or significant performance effects with existing applications. In one documented case, antivirus software identified a legitimate application as a virus and quarantined it, which generated a flurry of lawsuits.

3.5 CENTRALIZED SERVERS

Management consoles are available from antivirus vendors for performing centralized installation, configuration and operation of antivirus software running on clients connected on a computer network and obtaining and distributing engine and signature file updates. These highly automated tools are very useful for corporate IT departments who must manage and update hundreds or thousands of desktops and keep them current.

3.6 ICS CONSIDERATIONS

3.6.1 FALSE POSITIVES AND NEGATIVES

Not only do unknown and rapidly changing viruses escape pattern scanners, but relying on traditional virus scanning technology as the only antivirus strategy poses two additional problems.

- If a scan engine reports a file as being infected when in fact it is not, that reporting is considered a false positive. Depending on the removal techniques used (see Section 3.2), serious consequences to an ICS can occur. Because the number of viruses is increasing, the potential for false positives reported in scanning for known virus signatures also increases. These factors also add to the time and processing power required to complete virus scans.

- A false negative occurs when an antivirus scan engine misses a real infection. False negatives are a larger concern in using pattern-matching technology. A false negative is when a new or unknown virus does not match any of the signatures in the pattern scanner database file. The

pattern scanner does not detect these viruses and allows them to enter the system and execute their payloads.

3.6.2 PERFORMANCE IMPACTS ON ICSS

The addition of antivirus applications to a computer system requires that additional processing and memory resources be used. Although vendors of antivirus software strive to keep use of resources within acceptable limits, performance effects and general compatibility with an ICS's core applications should be assessed. A user must understand all options associated with the functionality of antivirus software such as heuristic analysis and signature updating and the possible effects they may have on system performance. In addition, ICSs are typically comprised of many workstations and servers working together to control a process. Performance degradation on one system could affect the performance of the overall system. Section 4 provides guidance for using antivirus software on industrial control systems.

4 GUIDELINES FOR USING ANTIVIRUS SOFTWARE ON ICSs

Information was gathered from industry end-users and vendors currently integrating antivirus software into their systems including control system configurations, needs and priorities for performance, and current practices and problems using antivirus software. The guidelines below were developed based on information collected from industry and laboratory testing.

4.1 INDUSTRY PRACTICES

Control system managers and engineers are apprehensive that a new signature or engine update will "break" a control application. After the IT department has completed their testing, many control departments perform additional pilot trials or screenings on new updates using offline or test systems. Rarely will a control system department allow the IT department to automatically update their workstations and servers, as the IT department frequently does with hundreds or thousands of business use clients.

Because of the safety, time criticality, reliability and availability requirements of control systems, control system departments do not generally use centralized management servers furnished with corporate additions. Instead, configurations and changes tend to be made locally by technicians or other control system department personnel on a client-by-client basis.

Some control system departments use a centralized update server to get control computer signature updates (in a process separate and distinct from the business IT updates). Some use manual "sneaker-net" or manual FTP to distribute these updates. In most cases, if there is an internal firewall separating the control systems from the business IT systems, an approved method for getting the engine or signature updates across this firewall is developed. This may be anywhere from a manual FTP to an automatic method using a distribution server in a "DMZ" between the IT and control networks.

In general, the project team found that control system departments have these requirements when considering automated tools for configuration and update processes:

- Ability to authorize/perform engine and signature updates on demand (vs. automatically)
- Provision to screen engine and signature updates before distribution and activation on clients
- Ability to authorize/perform manual scans upon local control

Note: An antivirus update distribution server was hooked up experimentally in an ICS lab setting. Several problems ensued, among them the inability to schedule or authorize when signature downloads occur. This particular distribution server was designed for typical IT environments where automatic features such as pushing signature updates are favorable to security administrators and end-users.

4.2 GUIDELINES

Caution: Always consult vendors of control system software and antivirus software for any available guidance. Any vendor guidance specific to control system and antivirus software compatibility should be given precedence over this generic guidance and test methodology.

4.2.1 PLANNING AND PURCHASING

- Similar to any security appliance, if antivirus software is installed, configured, or maintained by a separate IT group outside the expertise of a control systems operation group, both parties must work together closely to realize that control systems should not be treated as conventional IT systems due to their inherent critical timing constraints.

- Installation of antivirus software does not guarantee that your system will be safe from viruses; however, operational and maintenance policies pertaining to antivirus will help reduce the risk of a virus infecting you computer or network.

- Often, selection of an antivirus software package for use on a company's industrial control platform may be driven by corporate level selection for the business network. If this is the case, leverage the availability of corporate level expertise, keeping in mind the special requirements of the ICS.

- Ensure that the antivirus software package to be selected is certified to run on your control system workstation or server operating systems and currently installed service packs.

- Check if the vendor of your control system software recommends and supports the use of a particular brand of antivirus. In some cases, control system vendors may have performed regression testing across their product line for supported versions of antivirus software.

- Ensure that your control system workstation or server has adequate processor speed and memory to support both the antivirus software and resident control applications. Meeting these platform requirements does not insure that implementing antivirus will not affect the performance of your control system.

- Check for compatibility issues between the selected antivirus software package and other security technologies being used on the control system.

4.2.2 INSTALLATION AND CONFIGURATION

- Obtain all available installation and configuration guidance from the control system vendor if the vendor supports particular brands of antivirus software. Review all application installation and configuration documentation from the antivirus software vendor.

- Keep throttling settings to minimum by default. A configurable throttling feature is often available for manual scanning. As this configuration setting is increased, more of the workstation or server CPU time is given to the antivirus software process. This means that less time is given to concurrently running control processes. This can result in serious performance problems. It has also

been noted that these throttling settings are not always linear and may shift the majority of CPU processing time to the antivirus process at mid-level settings.

- When configuring file exclusion lists, determine what control application files should not be scanned during production time because of possible control system malfunction or performance degradation. Some control system vendors that support antivirus software will specify particular control files not to scan in their user documentation.

  Note: *One end-user reports exempting entire classes of data that if scanned may impact control system performance.*

- Carefully consider configuration settings and the effects that they may have on control system operation. Settings are available for scanning, removal, and handling viruses once detected. Read all antivirus software documentation for a description of each setting and consult with control system vendors for recommended antivirus software settings.

- Perform static testing to ensure that the installed antivirus does not impact your control applications.

- After successful installation and configuration of the antivirus software perform a full system scan before starting the control system.

4.2.3 SCANNING

- Schedule manual scans during control system down times or non-peak operational modes since manual scanning operations often cause control system performance impacts. If a system operates continuously and cannot be subjected to periodic manual scans, consider a procedure of running thorough manual scans on systems during down times and running in active scan mode during continuous control operation to the degree tolerable by the system. Contact your control system vendor and antivirus software vendor to identify risks associated with using the antivirus software in this way. It has been reported that some active scanners have sometimes had poorer detection capabilities than their manual scanner counterparts using identical test sets.

- Stagger manual scans across system servers and ensure that they are scheduled during non-peak operational modes. Special care should be taken to ensure that manual scan operations on redundant control servers are performed at different times.

- Minimize throttling settings during scheduled manual scans where the control system is in any type of operational mode. Increasing the throttling setting can cause substantial control system performance impacts.

- Determine any adverse effects resulting from the process of scanning a file. Refer to control vendor guidance for file exclusion guidance or consult with the ICS vendor before adding files to an exclusion list. Antivirus vendors warn that it is a serious action to put a file on an exclusion list and it should only be done in exceptional cases. Consider the following:

  - Data files usually pose no threat other than corruption of data when infected by a virus and sometimes present false-positive virus detection. Scanning these large and frequently accessed data files such as those associated with data historians could cause serious performance

degradation and if quarantined or cleaned, could adversely affect the operation of a historian application. Consider adding files of this type to the file exclusion listing to avoid scanning after consulting with the control system vendor.

- A file cannot be accessed or executed until scanning is completed. This could prevent a time critical process, such as an alarm, from executing on time or prevent subsequent response programs from running. Consult with the control system vendor for identification of any such processes. These may already be included on a control vendor scan exclusion list.

- Carefully consider the consequences of heuristic scanning configuration settings. Heuristic scanning techniques offer the capability to detect unknown or polymorphic viruses but are often prone to false positives. A typical antivirus software package also offers different levels of heuristic protection. In general, as the level of heuristic protection increases, so does the performance impact of the antivirus software.

4.2.4 REMOVAL

- Select methods for handling a infected files to minimize any adverse effects on the operation of a control system. Negative effects that could be encountered when handling infected files include:

  - Deleting an infected file that is critical to control system operation could disrupt operations, such as corrupting a historical database.
  - Quarantining a file may render files inaccessible and impact an operator's view of the ICS.

- Some end-users report restricting the method that antivirus software handles an infected files to operator alerts only which requires an operator to be available at all times to react to the detection of a virus.

4.2.5 MAINTENANCE

- Check the antivirus vendor web sites regularly for new virus definitions. New definitions should be downloaded and installed as soon as possible and in accordance with your organization's ICS security policy. Register with an antivirus vendor to receive notifications on new signatures as well as software updates.

- Use a local server to host new signature distribution. Some end-user and vendors of control systems report using a server located on a DMZ between the enterprise network and the process control network. Allowing direct access to the antivirus vendor's web site from the control system network is not advisable.

- Schedule virus definition updates during non-peak hours since this process often negatively impacts control system performance. Some end-users and vendors of control systems do not allow the use of automated schedulers and require that control systems pull updates from a server rather than having them pushed manually by an administrator or automatically by a server utility.

- Stagger the deployment of new signature files during slow periods when there is less control data traffic in order to avoid overloading the network.

- Consult with the antivirus software vendor concerning the availability of incremental signature updates. Incremental signature updates perform a partial update of a signature database based on its last revision.

- Perform off-line regression testing for all antivirus software patches, updates or new releases. New virus definitions should also undergo off-line testing before being released for distribution to control servers. There has been at least one occurrence of a faulty signature file released by an antivirus vendor that caused the systems hosting the antivirus application to malfunction.

- Conduct performance testing over all control system modes of operation expected during updates. Performing virus definition updates can cause control system performance impacts. Depending on performance test results, virus definition updates should be scheduled so they do not negatively impact operation of the control system. Use extreme caution if attempting to implement an automatic scheduler.

- Deploy new virus definition files to least-critical systems first to observe adverse reactions before they are applied to more critical systems in order to minimize the effects of a faulty signature file.

- Servers are available from antivirus vendors for performing centralized installation, configuration, operation, and reporting of antivirus software running on clients connected on a computer network. These utilities are often highly automated and designed for managing corporate antivirus installations. The stringent requirements for scheduling antivirus operations in an industrial control system environment may make these types of tools inappropriate for an ICS. It is recommended that responsible personnel talk to the antivirus software vendor and establish a test implementation to ensure the software performs as desired.

- Upgrading an operating system or installing operating system patches may cause the resident antivirus software to no longer function properly. When this is the case, consult with the antivirus software vendor for available patches.

5 ICS PERFORMANCE IMPACT TESTING

This section describes a test methodology for assessing the performance impacts caused by the addition of antivirus software to computer workstations and servers that host industrial control software. Scanning a system for viruses can have an adverse effect on overall system performance, as an antivirus engine competes with other applications for system resources. When performing active scans, most of today's antivirus scan engines are designed to distribute the additional overhead they introduce in the background and do not significantly impact the performance of other applications. In contrast manual scans often require enough of the available system resources for the antivirus engine to have a noticeable impact on the performance of other applications. Virus definition updates must be performed at regular intervals in order to maintain an up-to-date database of virus signatures. The process of updating these signatures often uses considerable system resources.

5.1 MEASURING PERFORMANCE

Assessment of industrial control system performance can include techniques such as monitoring a computer platform's performance variables, monitoring control system command and response communication packets, and analyzing the behavior of a control system from an operator perspective. There are many platform and control system parameters available to monitor, such as CPU usage, memory usage, and polling latency, as well as tools to perform the monitoring operation. In addition, an operator who is very familiar with the day-to-day performance of a system can identify sluggish, slow to respond effects caused by the addition of antivirus software. If the ICS component being addressed is a server, then additional performance characteristics such as a measure of the stability of communications with client machines should be monitored.

Assessment parameters, whether formal metrics or observations, should be carefully chosen to assure that the important factors critical to control system operation are captured. A listing of some of the performance monitoring tools used during the development of this test methodology can be found in Appendix B. Consider the following when choosing and monitoring performance variables:

- End-users should consult with control system vendors before installing any performance measuring tools on a control system platform due to compatibility and performance impacts that they in themselves may produce.

- In some cases performance monitors are already built into a control system product. In other cases, control system vendors may intentionally disable or block the execution of performance monitoring tools, such as those included with operating systems.

- There is a trade-off between frequency levels at which performance monitoring tools take readings; polling performance data too frequently can negatively impact system performance while polling too infrequently could yield results that are inaccurate or misleading. Infrequent performance polling can miss key control system performance data if the control system is sending or receiving communication messages at a higher rate than the performance tool polling rate.

- Identify computer processes stemming from the control system software, the antivirus software and any performance monitoring utilities. (Monitoring software process information can be used to assess the impacts these utilities have on the overall performance of the system being tested.)

- Run applications through all modes of operation in order to identify all processes associated with it. For example, running a virus definition update typically runs a process different from the scanning processes.

- Monitor resources remotely when possible in order to eliminate the additional CPU usage due to the addition of these applications on the workstation or server being tested.

5.2 TEST METHODOLOGY

This section presents a test methodology for assessing performance impacts introduced by antivirus software on control system performance. The test methodology is presented as a general set of test procedures based upon the different operational modes of antivirus software. Each test procedure recommends a method to test for performance impacts caused by the particular operational mode found in most antivirus software. Since control systems are implementation-specific, these test procedures should be used as a starting point when developing performance tests for specific control systems and may be expanded to support the specific configuration of the system under test.

The methodology consists of five sets of procedures to assess control system platform performance. Three of the procedures assess the major modes of operation of antivirus software: manual scanning, active scanning, and virus definition updates. In addition, procedures are provided for operational testing and for establishing a performance baseline. This methodology should be used on workstations and servers hosting industrial control software such as Microsoft Windows-based controllers, HMIs, and data historians, as well as control and SCADA servers.

The test methodology was developed based on information gathered from industry and research at NIST. Although development was performed using Microsoft Windows-based control platforms, the test methodology is general enough to be applied to other workstation and server platforms. The primary focus of this test methodology is performance testing.

Initial testing should be performed on a system that is not an active production system. Possible systems include corporate IT stations, development systems, off-line laboratory setups, production system simulations and production systems in down time.

End-users must decide to what extent they will follow this test methodology, depending on the conditions warranting testing. Some of these conditions include:

- First time integration of an antivirus software package
- Antivirus upgrades such as the installation of a new scanning engine
- Virus definition updates
- Production system reconfiguration

Performance testing should be implemented over the entire operational range of a control system including start-up, operating capacities and shutdowns (both standard and emergency). A test matrix for a system with different operating capacities is depicted in Table 2.

	Startup	Capacity 1	Capacity 2	Capacity 3	Normal Shutdown	Emergency Shutdown
Baseline (no antivirus software)						
Manual Scan						
Active Scan						
Definitions Update						
Engine Update						

Note: Engine updates should be treated as software patches.

Table 2: Example Test Matrix

5.2.1 ESTABLISHING A BASELINE

Generate a performance baseline for the control system component under test with no antivirus software running on the system. A performance baseline is the level of performance you can reliably expect during typical system usage and workload. Take the following into consideration when establishing a baseline of operation:

- A performance baseline should be established for each operational mode of the control system.
- It is important to be able to determine the peak loading of a system that normally occurs during startup or shutdown modes of operation. Care must be taken in this area since during critical times such as startup and shutdown, the file accesses are usually at a higher rate, meaning the antivirus software will be doing more work to verify the files are safe.
- Preferably, a baseline should be established prior to installation of the antivirus software to avoid use of any additional resources on the workstation or server.
- If qualifying a new antivirus package, the baseline could be drawn from how the system has performed with another antivirus vendor's qualified and properly configured package.

Procedure:

1. Select monitoring tools and identify all processes and performance variables. (Section 5.1)
2. Document baseline parameters with usage and workload conditions.
3. Run the control system applications.
4. If installed, the antivirus software should be set in an idle state. Check the system process listing for any antivirus application processes.
   Note: *Active scanning may need to be turned off since it is often activated in the default configuration. Also, deactivate any automatic enabler options.*

5. Collect data over typical control system usage and workload conditions over a period of time that insures all major control system processing events are captured.

5.2.2 STATIC TESTING

Static testing should be performed prior to the performance tests that follow. This procedure should be performed following initial installation and configuration of the antivirus software as well as the installation of maintenance updates. The tests include control system health and overall functionality.

1. Lock down the computer and turn off all unnecessary processes in accordance with the control system vendor recommendations.
2. Install or perform upgrade of antivirus software per vendor instructions.
3. Set antivirus software configurations for your control server.
4. Perform overall health checks on the concurrently running antivirus software and control system software.
5. Perform functionality testing on the concurrently running antivirus software and control system software.

5.2.3 MANUAL SCANNING

Compare control system performance during manual scanning operations with the control system performance baseline, taking the following into account:

- Identify for what periods of operation manual scans will be performed.
- Consider performing manual scans during non-peak operational periods.
- Look for performance deviations if priority settings are increased from minimum settings.
- Configure the antivirus software per any guidelines provided by the control system vendor.
- Follow the procedure below for each control system mode of operation.

Procedure:

1. Select monitoring tools and identify all processes and performance variables. (Section 5.1)
2. Disable active scanning. Also, deactivate any automatic enabler options.
3. Duplicate the data collection scenario used in establishing a performance baseline.
4. Configure a manual scanning operation on a group of directories and files located on the server hard drive and on peripheral drives.
   4.1. Total size of files to be scanned should be selected based on desired test time.
   4.2. Use many small files such as browser caches to force file open and read operations.
   4.3. Insert copies of the EICAR test virus throughout the directories of files to be scanned.
5. Trigger data monitoring and the manual scanning operation.
6. Collect data (e.g., total wall clock time and snapshot CPU utilization periodically) over the entire scanning operation.
7. Compare performance results with baseline performance data.

Note: If desired performance cannot be achieved, limit manual scanning operations to non-critical operational times and down times of the control system.

5.2.4 *ACTIVE SCANNING*

Compare control system performance during active scanning operations with the control system performance baseline. Active scanning operations should be performed over all control system modes of operation.

Procedure:

1. Select monitoring tools and identify all processes and performance variables. (Section 5.1)
2. Enable active scanning.
3. Test peripheral drive operations by dragging contents to the hard drive to trigger active scanning. Note: a system may be configured to prevent use of peripheral drives.
   3.1. Total size of files to be scanned should be selected based on desired test time.
   3.2. Use many small files such as browser caches to force file open and read operations.
   3.3. Insert copies of the EICAR test virus throughout the directories of files to be scanned.
4. Start all secondary applications that are used periodically during control system operation to force scanning of executable program files, as well as any associated DLLs.
5. Maximize network communication traffic of data coming onto the control server under test.
6. Collect data during steps 3, 4, and 5 above.
7. Compare performance results with baseline performance data.

5.2.5 *VIRUS DEFINITION AND ENGINE UPDATES*

Compare control system performance during virus definition or antivirus engine update operations with the control system performance baseline.

Procedure:

1. Select monitoring tools and identify all processes and performance variables. (Section 5.1)
2. Perform a virus definition update or engine update via the planned mechanism for distribution and installation.
3. Collect data during the virus definition or engine update operation.
4. Compare performance results with baseline performance data.

Note: Some engine updates require a system reboot.

5.3 TEST CASE OVERVIEW

A set of test cases was designed to accompany the test methodology and can be found in Appendix A of this document. These test cases are examples to demonstrate how to take the test methodology general procedures and generate application-specific, detailed procedures from them. The test cases also present users of this document with generic data and demonstrate some of the adverse ICS performance effects caused by antivirus tool configurations and practices.

6 CONTINUED EFFORTS

The results of this report including antivirus performance test procedures and industry good practices were presented at the June 2006 meeting of the Process Control Security Forum (PCSF) [11] in San Diego. Following this presentation an industry panel discussed the status of antivirus protection in the process control and SCADA communities. Industry panelists included antivirus vendors Symantec and McAfee, control system vendors Foxboro and Wonderware, and end-user Eastman Chemical. A positive audience response to the PCSF plenary panel was evident when over twenty-five conference attendees returned for a more intensive antivirus workshop the next day. Workshop discussions made clear that further interaction of end-users, control system vendors, and antivirus professionals would be valuable and lead to the formation of a PCSF interest group "Antivirus on Control Systems". The group plans to use the output of this project as a baseline in developing a common set of best practices and software requirements towards more efficient implementations of antivirus protection on industrial control systems. For more information or to join the PCSF "Antivirus on Control Systems" interest group, visit www.pcsforum.org/groups/77/index.php.

Appendix A: Test Cases

Test Case 1: Antivirus Software vs. Manufacturing HMI

Facility: NIST Industrial Control Security Test Bed [3]

Test System (see Figure 3):

Hardware:	Pentium III 1 GHz processor with 256 MB RAM Pentium II 450 MHz processor with 256 MB RAM Pentium II 266 MHz processor with 64 MB RAM
Operating System:	Microsoft Windows 2000 Professional – service pack 4
Antivirus Software:	not disclosed
HMI Software:	not disclosed
Most prominent processes:	HMI – HMI Software Antivirus – virus scanner Antivirus Update – virus definition update process

***Note**: Other related processes had only marginal impact on system performance.*

Tests were conducted on three separate PC workstations each running duplicate software setups. Each HMI application integration was configured with enough tag points to bring the total baseline processor load of each of the PC systems to between 40% and 60%. Wireshark, a packet monitoring software package, and Performance Monitor, a software tool included with Windows 2000 and XP to monitor system resources, were run on a separate computer to capture performance data of the servers. Computer platforms were chosen based on industry recommendations to address the typical lower end PCs considered the most vulnerable to performance impacts caused by the inclusion of antivirus software. The European Institute of Computer Antivirus Research (EICAR) test virus was injected through different test vectors, and communication packets were monitored between the HMI and the PLC.

Antivirus Software Setting Considerations:

Active Scanning

Used the default setting that scans files when accessed and modified. An alternative setting scans files only when they are modified.

Used the default heuristics setting. This setting sets the level of heuristic scanning to medium level (out of three available settings of low, medium or high). It is also possible to disable heuristic scanning. ***Note:** The antivirus software vendor recommends that the high setting only be used in cases where the*

user strongly suspects that a virus is already present on the system, because of the negative performance impact.

Manual Scanning

Performed testing over different throttling settings to observe performance impacts.

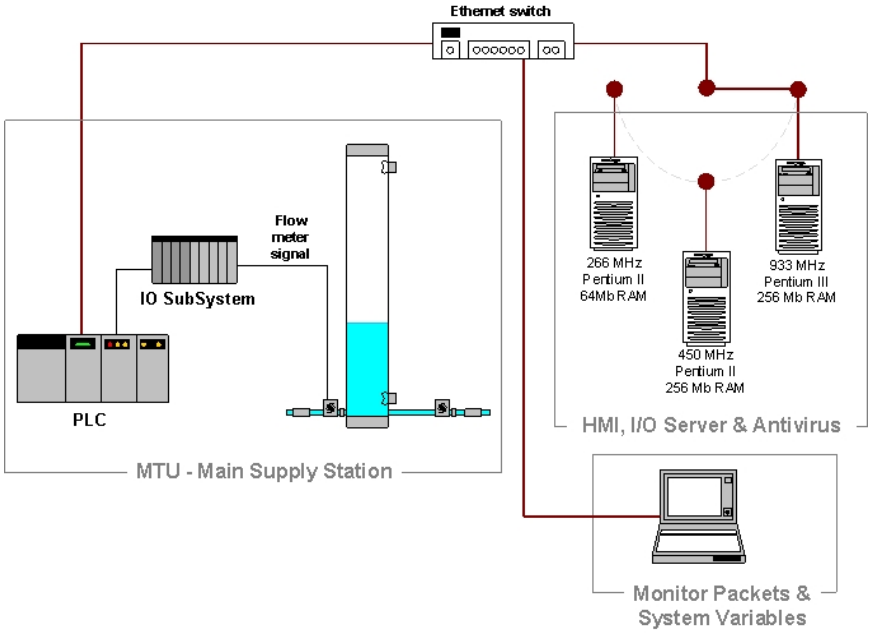


FIGURE 3: TEST BED SETUP ANTIVIRUS VS. HMI & I/O SERVER

Data collected:

Data included time, processor usage, percent used by each process, and communication packet information. The time between consecutive packets for monitoring a single sensor device was calculated and used as a measure of polling latency.
Note: *Memory usage was not monitored, but may be included in a subsequent version of this test case.*

Results and Analysis:

A manual scan test procedure was designed where a 100MB folder was created consisting of one thousand small files including copies of the EICAR virus. The folder was placed in the C: Drive of each test bed and a manual scan was executed only on the test folder. This procedure was carried out several times while adjusting the throttling setting on the antivirus software. This allowed observation of the range of effects a manual scan could have on the processor time of a control system relative to these settings. Results from this manual scanning procedure at the antivirus software's lowest and highest throttling settings are shown in Figures 4 and 5 respectively. Regardless of the throttling setting, once the manual scan began, the total processor time often reached the 80% to 100% range. As suspected, the antivirus software dominated the CPU time when set to the highest throttling setting and affected the performance of other processes like those belonging to the HMI software, which were reduced from the

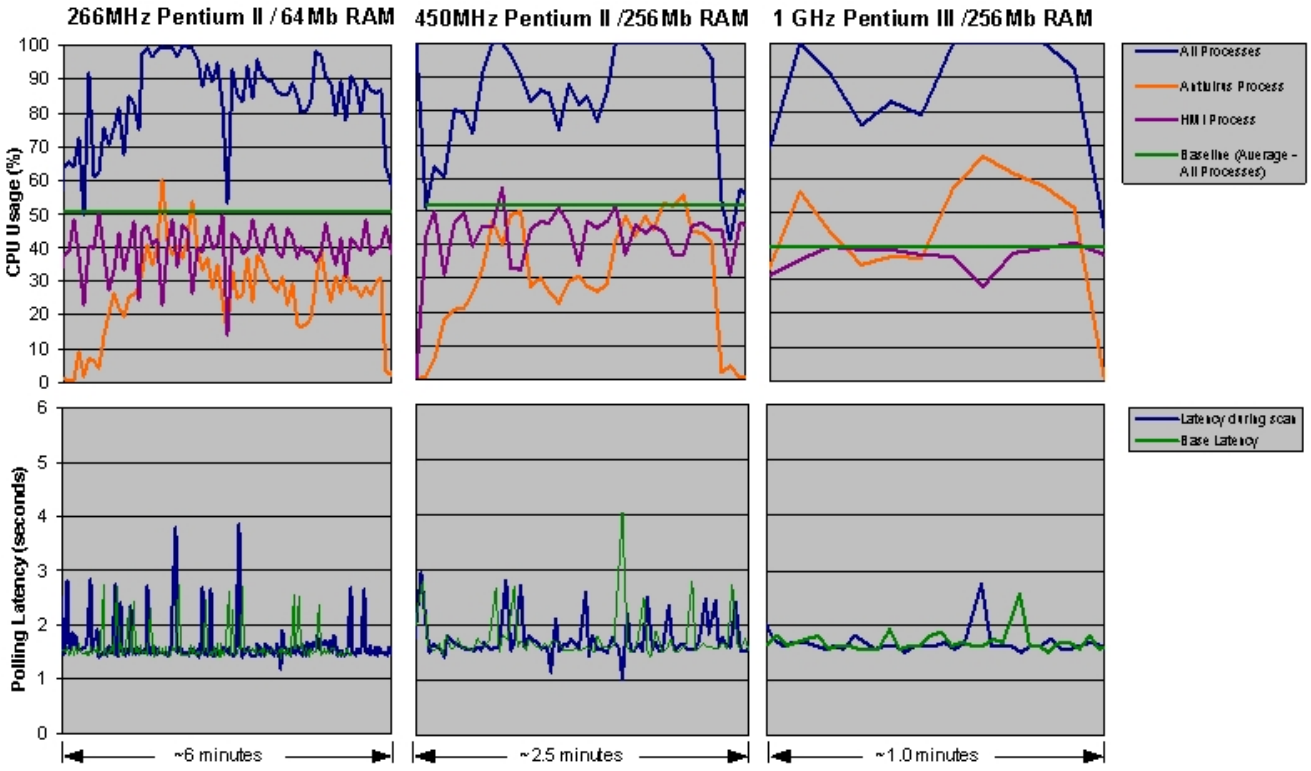


FIGURE 4: TEST CASE 1 DATA – MANUAL SCAN (LOW PRIORITY)

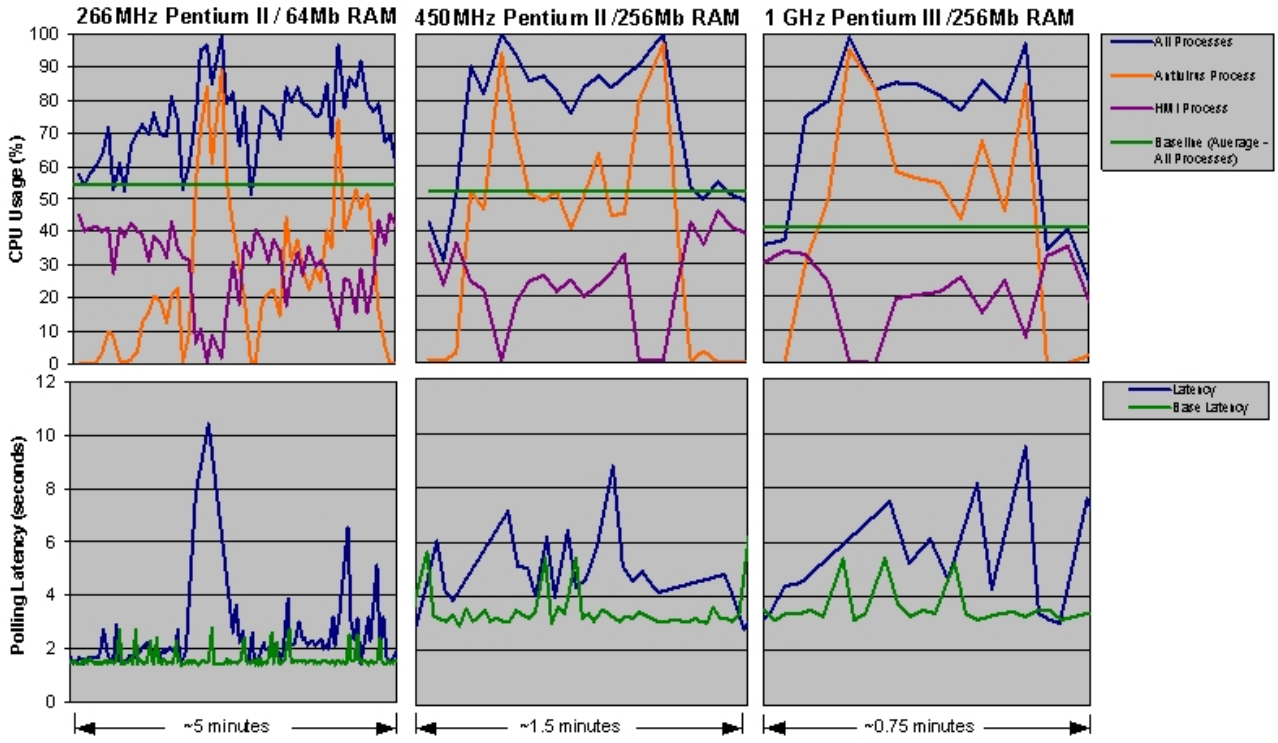


FIGURE 5: TEST CASE 1 DATA – MANUAL SCAN (HIGH PRIORITY)

40% to 60% CPU usages observed during baseline operations. When the throttling setting was low, the manual scanning process used the remaining processor time instead of taking precedence over the HMI software, keeping the antivirus software from hindering the performance of other processes.

There is a relationship between the antivirus software's throttling setting and the time required to run a manual scan. By increasing a throttling setting in the antivirus software, the time required to perform a scan was decreased. The scan duration was reduced by approximately 20% to 50% depending on the test bed that was being examined. The Pentium II 266MHz test bed displayed a 20% decrease while the Pentium III 1 GHz test bed displayed a 50% decrease. The highest throttling setting on antivirus software produced a significant impact on HMI performance. The lowest throttling setting produced a lower, more stable latency that mimicked the baseline data, while the highest setting produced a more chaotic latency pattern, usually with several relatively large spikes occurring during the manual scan. The average latency level of the high setting tends to be slightly greater than that of the low setting. This trend is most observed in the high priority data gathered from the Pentium II 266MHz test bed. There were no performance degradations from detecting and quarantining occurrences of the EICAR test virus.

Two different procedures were chosen to examine the effects of antivirus software actively scanning files in our control system. The first procedure involved transferring files from a removable hard disk to the hard drive and the second procedure involved transferring files from a floppy disk to a hard drive. There was a 100 MB directory on the removable hard disk and a 1.2 MB directory on the floppy disk, each containing many small files. Transferring data from these media to the test system hard drive enabled the active scanning process for the antivirus software. Data from the removable hard disk active scanning procedure is shown in Figure 6. The primary difference between the test and baseline CPU usages was due to the large numbers of file transfers associated with these active scanning procedures. Analysis of individual process CPU usage data determined that the antivirus processes dedicated to actively scanning files have a minimal impact on performance. This included any detection and quarantining of the EICAR virus. The time duration of the tests was primarily dependent on the transfer rates between the media drive and the hard drive. The effects on the latency within the control system's network while actively scanning accessed files were negligible. Latency appeared to average between 1.5 and 2 seconds for all test bed CPUs and all data media.

The virus definition update procedure was conducted on a local update file since there is no Internet connectivity from the NIST test bed. An executable file was used to update the virus definitions that could be manually placed on the control system computer. The executable file was obtained by downloading it from the antivirus software vendor's web site. While the virus definition updater was executed, the computers performance data was monitored and recorded. Data from this procedure can be found in Figure 7. Total and base processor times were basically the same except when the update utility was carrying out its operation. The update utility consumed relatively large amounts of processor time and caused the processor to reach its maximum operational capacity for the majority of the update. This occurred on all three platforms. Since the update utility dominates processor time, there is noticeable performance loss with the HMI processes and any other processes that might be running. This is most apparent on the Pentium II 266MHz system since its test duration was the longest. The latency was observed to remain in the base latency range of (1.5 to 2.5) seconds except while the update was being performed. The latency jumps to the (4.5 to 5) second range while the antivirus update process was running. The largest latency spikes were also observed on the Pentium II 266MHz platform.

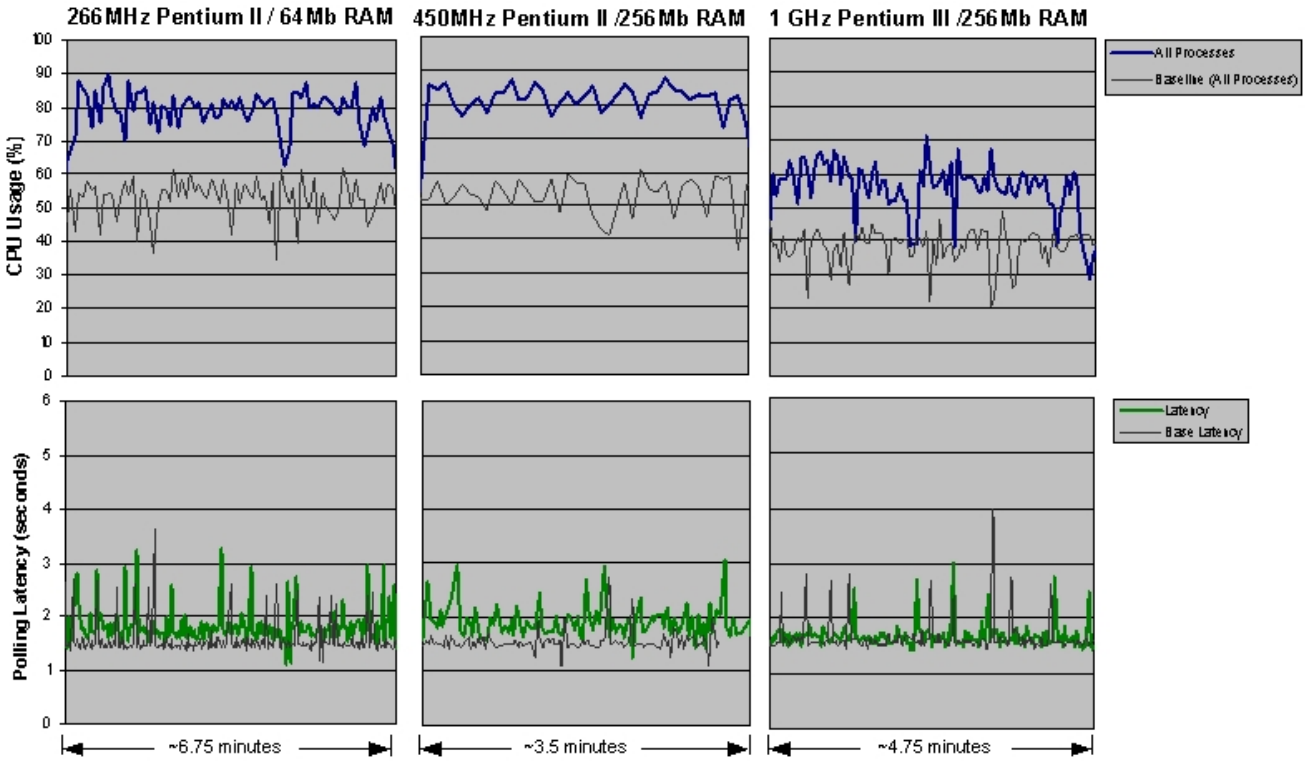
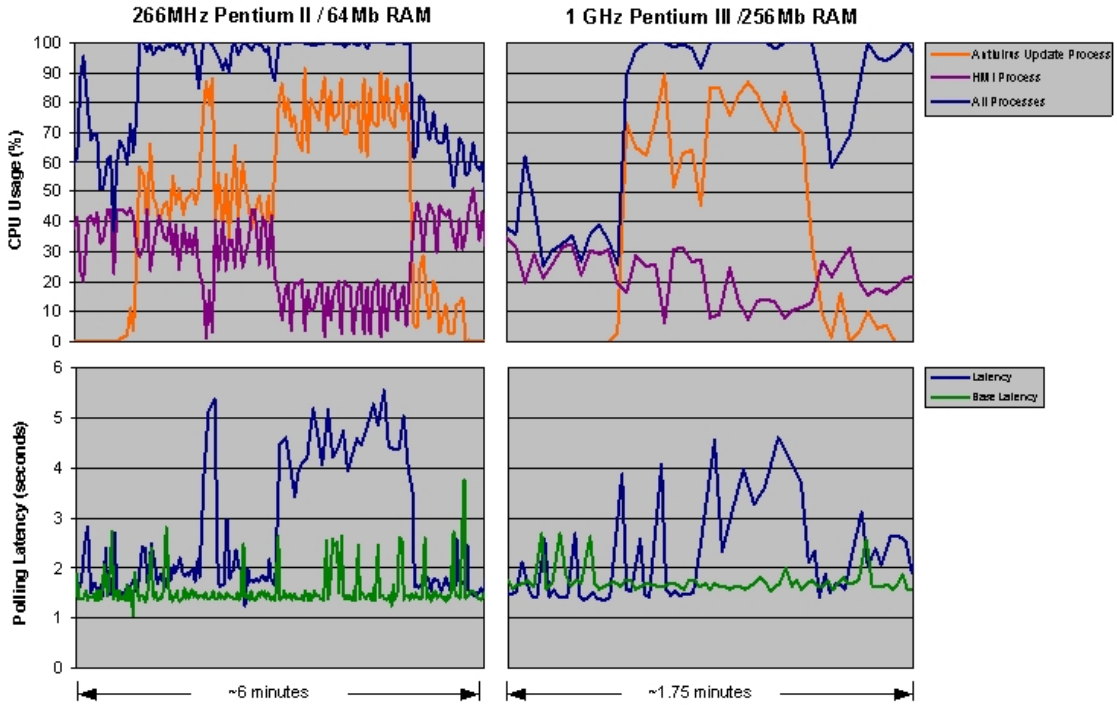


FIGURE 6: TEST CASE 1 DATA – ACTIVE SCAN



Note: Data for the 450MHz Pentium II was omitted due to suspected errors.

FIGURE 7: TEST CASE 1 DATA – VIRUS DEFINITION UPDATE

Test Case 2: Antivirus Software vs. SCADA HMI

Facility: Pacific Northwest National Laboratory (PNNL) SCADA Test Bed

Test System: *Hardware:* Intel Pentium IV 1400 MHz processor with 256 MB RAM

Operating System: Microsoft Windows XP Professional – Service Pack 2

Antivirus Software: not disclosed

SCADA Software: (HMI Component) not disclosed

Most prominent processes: SCADA1 – SCADA Software
SCADA2 – SCADA Software
Antivirus – virus scanner

Note: Other related processes had only marginal impact on system performance.

Antivirus Software Setting Considerations:

Active Scanning

Used the default setting that scans files when accessed and modified. An alternative setting scans files only when they are modified.

Used the default heuristics setting. This setting sets the level of heuristic scanning to medium level (out of three available settings of low, medium or high). It is also possible to disable heuristic scanning. *Note:* The antivirus software vendor recommends that the high setting only be used in cases where the user strongly suspects that a virus is already present on the system, because of the negative performance impact.

There was an option to scan in-memory threats. This option was disabled by default and left disabled for the tests. However, as the SCADA program loads and unloads data in and out of RAM, if this option was enabled system performance would most likely decrease.

Manual Scanning:

Performed testing over different throttling settings to observe performance impacts.

Expanded Threats

Optional detection of spyware, adware, dialers, and remote access software was disabled (default setting).

Note: *The more signatures the antivirus software is searching for, the more CPU cycles will be consumed looking for these extended threats.*

Data collected:

Data included time, memory usage, processor usage, and percent used by each process.

Results and Analysis:

The test results provided valuable insight into the implementation of antivirus software on SCADA systems. The manual scan throttling option in the antivirus software configuration was the single most important factor observed when comparing performance. At high priority, the SCADA system was usable, but extremely sluggish, and very noticeable to the user of the SCADA software while the scan was in progress. There were times when the SCADA software would take a second or two to respond to a mouse click or key press as the system became bogged down scanning for viruses. Although sluggish, the system remained stable, though the SCADA software decreased significantly in performance.

Data collected during this test can be found in Figure 8. Also, even when scanning on low priority the system was still noticeably sluggish. After observing the data and graphs, one should note that even with the antivirus software on the lowest priority setting, it still takes up more CPU cycles than all other processes combined. This is a very important factor to consider when dealing with time-critical SCADA systems, and reinforces the importance of limiting scans to off-peak times.

The tests performed show that active scanning does not have a major impact on system performance, although it is noticeable and may be a larger problem on older or less powerful systems. Note that active scanning was enabled during manual scanning operations.

Analysis of memory usage data concludes that "swap space" was not a critical factor – the memory usage rarely rose above 50%, so virtual memory was not needed on this test system under these conditions. However, if a computer with less available RAM is used, or more complicated or more resource-intensive software is used on the machine, this would be a very significant issue. Including memory usage in the test plan would allow those doing the testing to be aware of these critical factors.

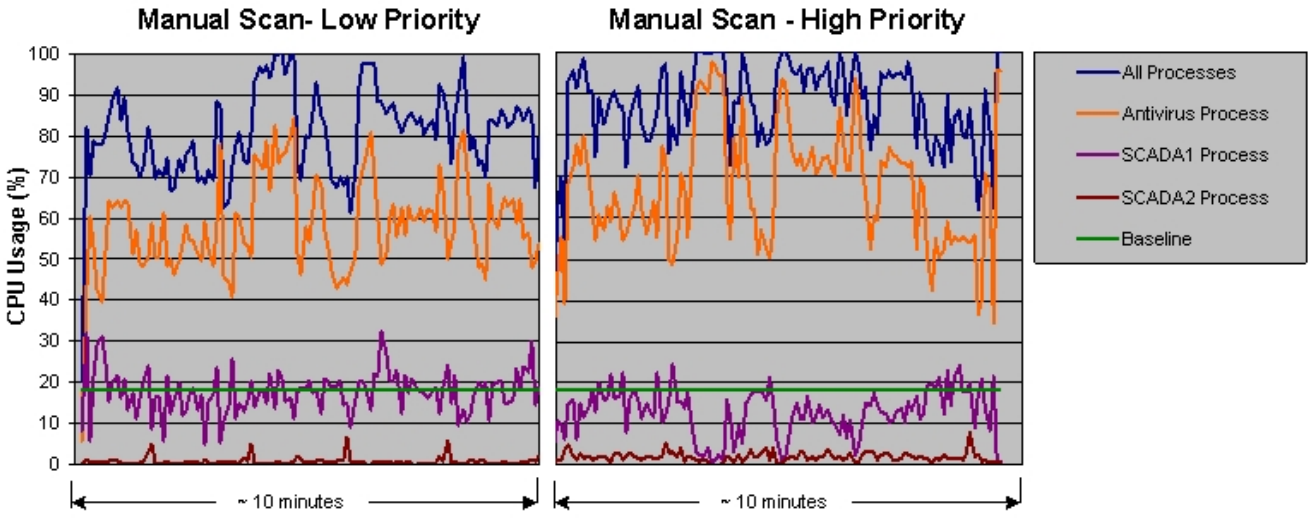


FIGURE 8: TEST CASE 2 DATA

Appendix B: Performance Tool Listing

Microsoft Task Manager

Description: A software utility included with Microsoft Windows.

Note: Task Manager may be blocked by some of the DCS vendors.

Use: Identify processes associated with applications.

Microsoft Performance Monitor

Description: Performance Monitor is a Microsoft tool for monitoring system performance. The application, included with Windows NT, 2000 and XP, can be configured to write data to a file in spreadsheet format and can also be configured to flag exceeded performance thresholds.

Use: Monitor system resources for depletions caused by the addition of antivirus software.

Resources:
http://support.microsoft.com/default.aspx?scid=kb;en-us;248345 (Performance Logs)
http://support.microsoft.com/default.aspx?scid=kb;en-us;244640 (Performance Alerts)

Wireshark

Description: Wireshark is a freely available, open source network protocol analyzer. It runs on all popular computing platforms, including Unix, Linux, and Windows.

Use: Monitor communication packets to and from the control system server running the antivirus software to measure data polling latencies caused by the addition of antivirus processes.

Resources: http://www.wireshark.org/

EICAR Test Virus

Description: The EICAR test virus is a non-viral test file that antivirus software will react to as if it were a virus. The file is a legitimate DOS program, and prints "EICAR-STANDARD-ANTIVIRUS-TEST-FILE!" when run. The EICAR test virus is available as a text file and two archived Zip files to test for viruses in multiple level archives. The test virus is constructed with the following 68 character (68 byte) string:

X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*

Use: Inject the EICAR test virus into the server being tested to check for performance degradation during the detection and disinfection stages of antivirus software execution.

Resources: www.eicar.org

Appendix C: Acronyms and Abbreviations

Selected acronyms and abbreviations

COTS	Commercial Off-the-Shelf
CPU	Central Processing Unit
DCS	Distributed Control System
DHS	Department of Homeland Security
DLL	Dynamic Link Library
DMZ	Demilitarized Zone
DOE	Department of Energy
EICAR	European Institute of Computer Antivirus Research
FTP	File Transfer Protocol
GHz	Gigahertz
HMI	Human Machine Interface
ICS	Industrial Control System
IT	Information Technology
LAN	Local Area Network
MB	Megabyte
MHz	Megahertz
NIST	National Institute of Standards and Technology
NSTB	National SCADA Test Bed
PNNL	Pacific Northwest National Laboratory
RAM	Random Access Memory
SCADA	Supervisory Control and Data Acquisition
SNL	Sandia National Laboratories
USB	Universal Serial Bus

Appendix D: Glossary of Terms

Selected terms and definitions

Active Scanning	An **antivirus** scanning option that operates in the background, using less memory resources than **manual scanning**. Active scanning examines files, diskettes, and system areas for **viruses** when information is accessed or changed. Active scanning is a serial process, preventing any application to start or file to be used prior to completion of the scanning process on it. Other names used for active scan include *real-time scan* and *on-access scan*.
Antivirus	Software products and technology used to detect malicious code, prevent it from infecting a system, and remove malicious code that has infected the system.
Backdoor	An undocumented way of gaining access to a computer system. A backdoor is a potential security risk.
Control Center	An equipment structure or group of structures from which a process is measured, controlled, and/or monitored.
Control Loop	A combination of field devices and control functions arranged so that a control variable is compared to a set point and returns to the process in the form of a manipulated variable.
Control Network	Those networks of an enterprise typically connected to equipment that controls physical processes and that is time or safety critical. The control network can be subdivided into zones, and there can be multiple separate control networks within one enterprise and site.
Control Server	A server that hosts the supervisory control system, typically a commercially available application for a **DCS** or **SCADA** system.
Control System	A system in which deliberate guidance or manipulation is used to achieve a prescribed value for a variable. Control systems include SCADA, DCS, PLCs and other types of industrial measurement and control systems.
Controlled Variable	The variable that the control system attempts to keep at the set point value. The set point may be constant or variable.
Controller	A device or program that operates automatically to regulate a controlled variable.
Cycle Time	The time, usually expressed in seconds, for a controller to complete one control loop where sensor signals are read into memory, control algorithms are executed, and corresponding control signals are transmitted to actuators that create changes in the process resulting in new sensor signals.
Data Historian	A centralized database supporting data analysis using statistical process control techniques.
Distributed Control System (DCS)	In a control system, refers to control achieved by intelligence that is distributed about the process to be controlled, rather than by a centrally located single unit.
Enterprise	An organization that coordinates the operation of one or more processing sites.
False Positive	A positive test result when the attribute for which the subject is being tested is reported but does not actually exist in that subject. For example, a false positive is said to occur if antivirus software generates an alarm because it finds a **virus** (a positive), but the file scanned does not contain a **virus** (false).

False Negative	A negative test result when the attribute for which the subject is being tested actually exists in that subject but is not detected. For example, a false negative is said to occur if **antivirus** software does not generate an alarm when a file scanned contains **malware**.
Field Device	Equipment that is connected to the field side on an ICS. Types of field devices include RTUs, PLCs, actuators, sensors, HMIs and associated communications.
Field Site	A subsystem that is identified by physical, geographical, or logical segmentation within the ICS. A field site may contain RTUs, PLCs, actuators, sensors, HMIs and associated communications.
Firewall	An inter-network gateway that restricts data communication traffic to and from one of the connected networks (the one said to be "inside" the firewall) and thus protects that network's system resources against threats from the other network (the one that is said to be "outside" the firewall).
Heuristic Analysis	A technique that looks for patterns, activities or suspicious code that may indicate a new **virus**. Most leading **antivirus** packages incorporate a heuristic analysis technique to detect new or previously undetected **in-the-wild viruses**.
In-the-Wild Virus	A **virus** that has been reported by at least two separate reporting agencies to Wildlist.org. The "Wildlist" is comprised of **antivirus** experts throughout the industry to maintain a *snapshot* of current infections. For a virus to be considered in-the-wild, it must be spreading as a result of normal day-to-day operations on and between the computers of unsuspecting users.
Human Machine Interface (HMI)	The hardware or software through which an operator interacts with a controller. An HMI can range from a physical control panel with buttons and indicator lights to an industrial PC with a color graphics display running dedicated HMI software.
Internet	The single, interconnected, worldwide system of commercial, government, educational, and other computer networks that share the set of protocols specified by the Internet Architecture Board (IAB) and the name and address spaces managed by the Internet Corporation for Assigned Names and Numbers (ICANN).
Malware	Software or firmware intended to perform an unauthorized process that will have adverse impact on the confidentiality, integrity, or availability of an information system. A virus, worm, Trojan horse, or other code-based entity that infects a host. Spyware and some forms of adware are also examples of malicious code (malware).
Manual Scanning	An **antivirus** scanning option that detects viruses using signature matching in selected groups of files and/or directories and may be initiated by a user or automatic scheduler. Manual scanning generally takes longer than active scanning since a large number of files are usually scheduled for scanning. Another name often used for manual scanning is *on-demand scanning*.
Master Terminal Unit (MTU)	See **SCADA Server**.
Operating System	An integrated collection of service routines for supervising the sequencing of programs by a computer. An operating system may perform the functions of input/output control, resource scheduling, and data management. It provides application programs with the fundamental commands for controlling the computer.

Real-Time	Pertaining to the performance of a computation during the actual time that the related physical process transpires so that the results of the computation can be used to guide the physical process.
Redundant Control Server	A backup to the **control server** that maintains the current state of the **control server** at all times.
Remote Terminal Unit (RTU)	A computer with radio interfacing used in remote situations where communications via wire is unavailable. Usually used to communicate with remote field equipment. PLCs with radio communication capabilities are also used in place of RTUs.
Scan Engine	Software that scans computer systems for malware threats. Antivirus scan engines use virus signature files to receive updates on the latest security threats.
SCADA Server	The device that acts as the master in a SCADA system.
Signature Files	A frequently updated database containing thousands of signatures of known viruses. **Antivirus** vendors continuously collect **malware** specimens from which they derive a signature (fingerprint) of the **virus** to be added to this database.
Signature Matching	A technique that matches file code against the contents of a **signature file**. The matching process does not necessarily have to look at all the code in a file; for instance, many viruses leave the original file contents intact and append the **malware** code.
Spyware	Software that is secretly or surreptitiously installed onto an information system to gather information on individuals or organizations without their knowledge; a type of malicious code.
Supervisory Control and Data Acquisition (SCADA)	A generic name for a computerized system that is capable of gathering and processing data and applying operational controls over long distances. Typical uses include power transmission and distribution and pipeline systems. SCADA was designed for the unique communication challenges (delays, data integrity, etc.) posed by the various media that must be used, such as phone lines, microwave, and satellite. Usually shared rather than dedicated.
Threat	Any circumstance or event with the potential to adversely impact agency operations (including mission, functions, image, or reputation), agency assets, or individuals through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service.
Throttling	An **antivirus** configuration setting for manual scanning used to reserve more CPU time for scanning process in order to shorten the time to complete a **viru**s scan.
Trojan Horse	A computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the program.
Virus	A hidden, self-replicating section of computer software, usually malicious logic, that propagates by infecting—i.e., inserting a copy of itself into and becoming part of—another program. A virus cannot run by itself; it requires that its host program be run to make the virus active.
Virus Definitions	Predefined signatures for known malware used by antivirus detection algorithms.

Virus Definition Update	A maintenance aspect on **antivirus** software where new **signature files** are downloaded from a vendor web site at regular intervals and updated on the computer systems running **antivirus** software. Also called *signature file update*.
Vulnerability	Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source.
Worm	A computer program that can run independently, can propagate a complete working version of itself onto other hosts on a network, and may consume computer resources destructively.
Zero-day Exploit	An exploit (e.g., malware) that takes advantage of a security vulnerability on the same day that the vulnerability becomes generally known or earlier.
Zoo Virus	A virus that is carefully contained in libraries maintained by antivirus researchers that exists for testing and has never infected a real world computer system.

Appendix E: References

[1] DOE, DHS, *Roadmap to Secure Control Systems in the Energy Sector*, 2006, http://www.controlsystemsroadmap.net/pdfs/roadmap.pdf.

[2] Department of Energy (DOE), National SCADA Test Bed (NSTB), http://www.ea.doe.gov/pdfs/nstb_factsheet.pdf.

[3] NIST Industrial Control Security Test Bed, http://www.isd.mel.nist.gov/projects/processcontrol/.

[4] Falco, et al, *IT Security for Industrial Control Systems*, NIST Internal Report, 2002, http://www.isd.mel.nist.gov/documents/falco/ITSecurityProcess.pdf.

[5] Stouffer, Falco, NIST SP 800-82, *Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security*, Draft, 2006, http://csrc.nist.gov/publications/nistpubs/800-83/SP800-83.pdf

[6] Mell, Perter, et al, NIST SP 800-83, *Guide to Malware Incident Prevention and Handling*, 2005, http://csrc.nist.gov/publications/nistpubs/800-83/SP800-83.pdf

[7] Microsoft Solutions for Security – The Antivirus Defense-in-Depth Guide, 2004, http://www.microsoft.com/technet/security/topics/serversecurity/avdind_0.mspx.

[8] Harley, Slade, Gattiker, *Viruses Revealed*, McGraw-Hill Companies, 2001.

[9] Peter Szor, *The Art of Computer Virus Research and Defense*, Symantec Corporation, 2005.

[10] The European Institute of Computer Antivirus Research (EICAR), http://www.eicar.org.

[11] Process Control Systems Forum (PCSF), http://www.pcsforum.org.

NIST Special Publication 1058

Using Host-Based Antivirus Software on Industrial Control Systems:

Integration Guidance and a Test Methodology for Assessing Performance Impacts

Joe Falco
Intelligent Systems Division
Manufacturing Engineering Laboratory
National Institute of Standards and Technology

Steve Hurd
Sandia National Laboratories

Dave Teumim
Teumim Technical, LLC.

September 2006

U.S. Department of Commerce
Carlos M. Gutierrez, Secretary

Technology Administration
Robert Cresanti, Under Secretary of Commerce for Technology

National Institute of Standards and Technology
William Jeffrey, Director

National Institute of Standards and Technology Special Publication 1058
Natl. Inst. Stand. Technol. Spec. Publ. 1058, 50 pages (September 2006)
CODEN: NSPUE2