Correlating Easily and Unobtrusively Queried Computer Characteristics to Number and Severity of Vulnerabilities

Jonathan M. Mercado
Brigham Young University

Correlating Easily and Unobtrusively Queried Computer Characteristics to Number and Severity of Vulnerabilities

Jonathan M. Mercado

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

School of Technology

Brigham Young University

ABSTRACT

Correlating Easily and Unobtrusively Queried
Computer Characteristics to Number
and Severity of Vulnerabilities

Jonathan M. Mercado
School of Technology, BYU
Master of Science

Cybersecurity has become a top-of-mind concern as the threat landscape expands and organizations continue to undergo digital transformation. As the industry confronts this growth, tools designed to evaluate the security posture of a network must improve to provide better value. Current agent-based and network scanning tools are resource intensive, expensive, and require thorough testing before implementation in order to ensure seamless integration. While surfacing specific vulnerability information is imperative to securing network assets, there are ways to predict the security status of a network without taking exact measurements. These methods may inform security professionals as to where the weakest points of the network lie quickly, unobtrusively, and cost-effectively. This thesis proposes a methodology for identifying correlations between host configuration and vulnerability, then specifically examines easily queried characteristics within the Microsoft Windows operating system that may be vulnerability predictors.

After taking measurements of forty hosts, it was discovered that there is a strong ($r > 0.80$) correlation between several metrics and total number of vulnerabilities as measured by the Tenable Nessus network scanner. Specifically, total number of open TCP ports ($r = 0.82$), total number of programs installed ($r = 0.90$), days since last restart ($r = 0.97$), and days since last windows update ($r = 0.93$) were found to be strong candidates for identifying high-risk machines. A significant correlation was also found when measuring the total number of logged in users ($r = 0.68$). Correlations were not as strong when considering subsets of hosts in similar environments. These findings can be used in tooling which will quickly evaluate the security posture of network hosts.

Keywords: cybersecurity, vulnerability, network

ACKNOWLEDGEMENTS

Thanks to Dr. Dale Rowe for his support through my undergraduate and graduate studies as well as Trey Clasen for supporting this research directly.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 Nature of the Problem

On December 7, 2017 Gartner forecasted that worldwide cybersecurity spending will reach $96 billion USD in 2018, up 8 percent from 2017. This comes as a result of increased awareness of the threats present in an online world and the continued push toward digital transformation that businesses recognize they must pursue in order to stay relevant. They also forecast that, "by 2020 more than 60 percent of organizations will invest in multiple data security tools … up from approximately 35 percent today" (Meulen and Pettey, 2017). As organizations increase their investment in cybersecurity, they will require technology that can be integrated effectively into their network environments. Today there are many challenges associated with such integration; additional innovation is required to develop practical and quickly assimilated solutions that maximize a return on investment.

1.1.1 Unseen Network Vulnerability

One of the challenges enterprises and other organizations face as their infrastructure grows is keeping track of the assets, dependencies, technologies, and users built into their ever-expanding networks. Without proper visibility it becomes increasingly difficult to evaluate potential vulnerabilities. Unfortunately, it is often the case that the low-hanging fruit of easily compromised systems is sufficient to affect widespread infrastructure infiltration despite security measures implemented in areas where network visibility is high.

Security professionals need tools that surface vulnerability data to provide actionable insight into the state of network resources. Without this information, the prioritization and implementation of security initiatives cannot be performed effectively. While a wide array of options exist that are designed to provide this visibility, they present significant hurdles that must be cleared in order to be of any benefit.

1.1.2 Agent-Based Solutions

Agent-based security software is typically built into infrastructure deployment such that any new resource on the network automatically reports to a centralized management platform. This approach not only allows administrators to keep track of assets and evaluate vulnerabilities, but can also provide real-time intrusion prevention, integrity monitoring, anti-malware, and other features. This method also front-loads integration work before it can be enabled at low risk to production systems. Testing must be done to ensure that agent activities do not interfere with normal operation. Vendor updates, hardware configuration changes, new security rules, and software bugs can all contribute to outages. Testing and evaluation, when done properly, can take months depending on the size and complexity of the production system. It must then be tuned, and the noise floor lowered sufficiently to provide vulnerability insights.

1.1.3 Network Scanning Solutions

An alternative to deploying software to every network resource is to perform network scanning from a central location. Network scanning software evaluates the security posture of hosts by probing each one from across the network. This methodology has the advantage of not requiring per-host installation, but the scanning platform and supporting resources must still be built and maintained. Such solutions, as with their agent-based counterparts, require significant testing and development as network scanning can affect system performance and even cause outages. Careful observation and tuning are needed to ensure thorough results without service degradation. In addition, scanning creates abundant noise on a network which may interfere with network devices. It may be flagged or blocked by other security technologies and require filtering, lowering the fidelity by which such technologies can effectively identify real threats.

1.1.4 Lack of Fast and Effective Tooling

If the Gartner forecast holds true, the adoption of cybersecurity tools applied to networks already in existence will continue. Once an organization decides it is time to properly evaluate and protect network assets, they will likely turn to agent-based or network scanning solutions as described to obtain a detailed picture of their security posture. While these methodologies provide a plethora of useful data, neither allows an availability-sensitive network in dire need to be assessed quickly. The industry needs safe and effective ways to find the most vulnerable areas so that spot cleaning can be performed while more comprehensive technologies are ramping up integration. Before these tools can be developed, research must be done to reveal strategies such tooling can employ to competently identify weaknesses. There are many targets throughout a network that should be addressed in this research. Databases, network devices, workstations and other exploitable technologies would benefit from tools that could quickly and unobtrusively provide best-guess assessments of potential vulnerability.

1.1.5 Problem Statement

The cybersecurity industry lacks quick and unobtrusive methods to evaluate potential vulnerabilities within a network.

1.2 Research Approach

This study is an initial step toward identifying methods that will quickly and unobtrusively identify network resources that are likely vulnerable to network attacks without directly measuring the vulnerabilities themselves. These methods are not designed to stand in for a thorough investigation, but instead seek to reveal the weakest targets which, if found by an adversary, may lead to total network compromise.

In this study we take a look at potential risk markers found in home computers, shared or public computers, and business workstations; all of which are network edge devices. While network infrastructure can be controlled and maintained centrally within an organization, employees and other users work on network edge devices that are susceptible to error and mistakes make by less security-minded individuals. Following is an assessment of potential vulnerability in the Microsoft Windows operating system by collecting a series of easily-queried characteristics and ascertain their correlation with the number or severity of actual vulnerabilities on a system. While the scope of this assessment excludes server platforms, lessons learned from the results obtained can also inform what might be expected in server environments. Ultimately this research outlines a methodology for discovering useful correlations and implements it on a number of measurable computer characteristics. This provides a first step toward the development of technology that could more quickly and unobtrusively grant insight into the security posture of a network.

The purpose of this study is to implement my new methodology to identify characteristics in home computers, shared computers, and workstations on a network that correlate to significant vulnerability. This commences with a review of current cybersecurity recommendations and best practices for edge-device users. The easily-queried characteristics that are collected will correlate with these recommendations to provide a point of reference as to their efficacy using objective data. Should adherence to these recommendations correlate with low counts and severity of actual computer vulnerability, then there is evidence to support them. If not, there is reason to reevaluate assumptions regarding what behaviors keep users secure. The hypothesis is that computer characteristics exist that can be quickly and unobtrusively queried which significantly correlate with number or severity of vulnerabilities.

1.3 Thesis Outline

In the following chapter an analysis is performed of related literature including cybersecurity recommendation documents and publications that demonstrate work that has been done to improve the performance and accuracy of vulnerability assessment tools. Afterward the study methodology is described. An analysis of the results and any correlation discovered will follow a presentation of the data collected. Lastly, implications of the results and challenges faced during the study are discussed. Limitations in applicability will be highlighted as well as what can be done in the future to continue research in this area.

2 LITERATURE REVIEW

Two focus areas exist in the literature review. First, a discussion of relevant publications in related research with regard to vulnerability assessment and the techniques and ideas that attempt to maximize efficacy and efficiency. Second, an evaluation of publications that present themselves as cybersecurity best practices for end users. The suggestions presented therein are the basis for which metrics were chosen for measurement as part of the research. Finally, rational for metric selection based on the commonly recommended actions and behaviors presented is discussed.

2.1 Related Research

A lot of work has been done to evaluate available network security tools in the context of the minimum needs of a particular organization. Material cost is the primary driving force in maximizing efficiency and efficacy of such tools as cybersecurity expenses fall under the category of "Cost of Goods Sold" from a financial perspective. In other words, no direct value to the bottom line is inherent in cybersecurity expenditure. The logic follows that a business of any sort would seek to only meet the minimum bar required to mitigate egregious cybersecurity risk and spend as little money as possible in doing so.

For example, Sawik demonstrates a "mixed integer programming approach" to automating the decision-making process for which appropriate cyber-threat countermeasures should be employed in a particular network to be cost-effective. His approach takes into consideration not only the initial cost and effectiveness of various solutions, but the probability that particular attacks will take place as well as potential losses from successful attacks against infrastructure. (Sawik, 2013).

As cloud-based services continue to increase in popularity, securing infrastructure in the hyper-scale context is also being widely discussed. Patel et al. performed a comparison of traditional intrusion detection and firewall systems to other solutions that are more effective in a cloud scenario. Many metrics were discussed which are pertinent to this research including efficiency, scalability, resource intensiveness and complexity. These comparisons are directly applicable and very helpful in making solution choices (Patel et al., 2013). Again, this is intended to provide the most value for those implementing security solutions when cost is an important factor of the analysis. Future such analysis would do well to include security solutions based on vulnerability correlations.

One of the more novel methodologies that has been proposed and evaluated for detecting vulnerability and vectors for potential compromise uses techniques involving attack graphs. Phillips & Swiler showed that network vulnerability analysis can be performed by creating a logical progression of attack given an understood initial set of possible attack vectors. Risk is determined based on the likelihood and severity of those attacks on a specific network provided its topology. The merit of this technique as it is not necessarily evaluating vulnerabilities, but is instead taking a network state and intuitively predicting likely scenarios. It is making the same kinds of predictions that this thesis attempts to demonstrate, but is congregating significantly more data and drawing conclusions to the scale of the entire network. Of note is the requirement to collect attributes about each host to be fed into the model. The authors list the attributes specifically for every device on the network. They are given as follows:

> "1. Machine class: workstation, printer, router, etc.
> 2. Hardware type: e.g., SUN SPARCstation^{TM}
> 3. Operating System
>   a. O.S. patches that have been installed.
> 4. Users (Initially just the classes of users, i.e. root, normal, privileged.)
> 5. Configuration
>   a. Ports enabled
>   b. Services enabled and privilege level on the services
>   c. Any intrusion detection applications installed
> 6. Type of network(s) the device is on (Ethernet, FDDI, ATM, etc.)
> 7. Physical link information such as type of communications media"

The information required looks remarkably close to the easily and unobtrusively queried computer characteristics that are discovered as part of this thesis. The difference here is that this data is being collected to inform the graph-based model to make its predictions. This study considers the data as being useful in and of itself as a tool to predict vulnerability at a host level (Phillips & Swiler, 1998).

Expanding on the idea of graph-based vulnerability assessment is a much more recent publication by Albanese, Jajodia, and Noel which again addresses the motivation of reducing cybersecurity related overhead as much as possible. In this article a number of suggestions are made to make the attack-graph methodology more effective. One improvement involves specifying which corrective actions are allowed to be made taking cost and impact into account (Albanese, Jajodia, and Noel, 2012). If there are multiple potential solutions to mitigating risk, take the path of least resistance. This is the kind of reasoning that gets organizations to a more secure state without limiting action to common methods that are inefficient to implement and expensive to operate.

Viduto et al. look at cost-effectiveness and finding the correct solutions for the organization in question. They present a couple of approaches and propose an eleven-step process they call the “Risk Assessment and Optimization Model (ROAM)”. The article also touches generally on a large body of work that is being done in this area, attempting to discern the best course of action an organization should take from different perspectives. They discuss direct approaches and heuristics, including measuring residual vulnerability and cost to patch and measuring impact if vulnerability is exploited as methods for risk assessment. NIST and ISO guidelines are cited as being common but less effective ways to approach risk assessment and mitigation (Viduto et al., 2012). Again, generally the way to asses risk involves vulnerability discovery, a step that could be bypassed in order to get an easy, if imperfect, view of where the most vulnerable targets may lie.

In general, there have been many efforts made to discover better methods to secure networks that are less resource intensive and less expensive. The plethora of ideas that have been circulated touch on every part of the network stack in every part of the network. There are also a number of patents related to this research including *Inventory Management-Based Computer Vulnerability Resolution System* for automating remediation actions based on network visibility information and *Combining Assessment Models and Client Targeting to Identify Network Security Vulnerabilities* which outlines a strategy for vulnerability assessment which mixes client-side agents and network-based scanning to maximize performance (Banzhof et al., 2006) (Soderberg and Kachachi, 2012). Work has also been done to address a hosts ability to self-regulate while minimizing impact as in *Effective and Efficient Malware Detection at the End Host* (Kolbitsch et al., 2009). In all, there is a large and growing body of work addressing these issues. Finding the right balance in security investment continues to be top-of-mind for cybersecurity research.

2.2 Cybersecurity Best Practices

There are many companies and organizations willing to offer advice to users concerning cybersecurity best practices to users. Whether a user has followed these recommendations is evident in a host's configuration. If best practices documents are effective, these configuration characteristics will predict the vulnerability of a host. Therefore, this study selects computer characteristics for measurement which indicate compliance to common cybersecurity recommendations. To this end, a number of online publications targeted at end users of personal or business PCs were collected. These particular publications were selected on the basis of general visibility and availability to end users actively seeking to protect themselves technologically.

An online search of cybersecurity recommendations yields significant amounts of information. The selected results come from a variety of sources including universities, media outlets, government agencies, and security companies. Following is the list of publications:

- *Basic Computer Security* by How-to Geek
- *50+ Internet Security Tips and Tricks from Top Experts* by Heimdal Security
- *Top Ten Safe Computing Tips* by the Michigan Institute of Technology
- *10 Tips to Stay Safe Online* by McAfee
- *Top 10 Secure Computing Tips* by University of California Berkley
- *Cybersecurity for Small Businesses* by the Federal Communications Commission (FCC)
- *10 Basic Cybersecurity Measures* by the Water Information Sharing and Analysis Center (WaterISAC)
- *CIS Controls List* by the Center for Internet Security (CIS)

- *10 Ways to Develop Cybersecurity Policies and Best Practices* by ZDNet
- *The 5 Cybersecurity Practices Every Employee Should Follow* by the Washington Post
- *Top Ten: The Most Important Cyber Security Tips for Your Users* by Cisco
- *Tips That Will Save Your Digital Life from Getting Hacked* by Business Insider
- *Stop. Think. Connect.* by the Department of Homeland Security (DHS)
- *Cyber Tips* by the New Jersey Cybersecurity & Communications Integration Cell (NJCCIC)

As is evident from the titles, it is common to present cybersecurity recommendations as lists that make it easy to categorize the kinds of advice they are offering. There are, however, a number of differences between them. For example, the New Jersey Cybersecurity & Communications Integration Cell's *Cyber Tips* webpage provides significant information regarding the risks that are present before offering solutions. In contrast, MIT's *Top Ten Safe Computing Tips* site bullet-points their recommendations in quick succession. One of the more interesting presentation formats is found in Heimdal's *50+ Internet Security Tips & Tricks from Top Experts* which offers the top three cybersecurity concerns from each of nineteen professionals in the field.

Evaluating the information from the publications yielded several categories of recommendations. Categories were selected for this thesis where multiple publications had made similar recommendations and which taking measurements from host machines might provide some insight into their efficacy:

- Passwords – unique and strong passwords were recommended as well as automation tools
- Antivirus – general instruction was to have any antivirus installed and up-to-date
- Privilege – role-based access control and limited use of administrator privileges
- Software Updates – whether it be OS or application updates

- Firewalls – more emphasis on firewall existence and less on proper configuration
- Media Devices – limit or audit USB devices and/or use caution when plugging them in

Other very common recommendations included making backups, avoiding malicious hyperlinks, having a mobile device security plan, and performing employee training. These suggestions are not measurable in any meaningful way using the PowerShell 2.0 host information collection method described in chapter 3, and are therefore not included in this research. Table 1 summarizes the selected recommendation categories that each publication addresses.

Table 1: Cybersecurity Recommendation Categories by Publication

Publisher	*Passwords*	*Antivirus*	*Privilege*	*Software Updates*	*Firewalls*	*Media Devices*
How-To Geek	X	X		X		
Heimdal Security	X	X		X		
MIT	X	X		X	X	
McAfee	X			X	X	
UC Berkley	X	X		X	X	
FCC	X	X	X	X	X	
WaterISAC	X		X	X	X	
CIS		X	X	X	X	X
ZDNet				X		
Washington Post				X		
Cisco	X	X				X
Business Insider	X			X		X
DHS	X	X		X		X
NJCCIC	X	X		X	X	X
Total	**11**	**9**	**3**	**13**	**7**	**5**

3 METHODOLOGY

In chapters one and two we established the context behind the research, this chapter will demonstrate the approach used to discover vulnerability correlations.

3.1 Scope

As discussed in chapter one, there are many network devices and endpoints that would benefit from research that would reveal correlations between easily obtained characteristics and real vulnerability. For this study specifically, the scope is limited to the Microsoft Windows operating system as designed for personal computer endpoints. Supported versions include Windows 7 Home Premium, Professional, Enterprise, and Ultimate, Windows 8 and 8.1 Home, Professional and Enterprise, and Windows 10 Home, Professional, and Enterprise. PCs selected for testing included shared lab computers, personal devices, and workstations.

3.2 Easily and Unobtrusively Queried Computer Characteristics

In order to collect data from hosts, a Windows PowerShell script was written that could be run with administrative access on each machine in the study. For compatibility between all operating system versions in scope, the script uses only functionality available in PowerShell version 2.0. In the interest of finding direct correlations with limited variables, the characteristics measured are able to be represented by a simple Boolean or integer value. Given the categories of interest selected in chapter two, the determination was made to gather the following metrics from personal computers:

3.2.1 Password Metrics

Asking Windows about the state of passwords on a system is not straightforward. As with any system, the less it knows about the passwords of its users the better. Unfortunately, it is not possible to query for the length or complexity of passwords specifically. However, there are a couple of interesting metrics that allow us to understand if general password requirements were bypassed for convenience sake. Accounts that are not required to have a password of some sort, and whether the auto-logon feature has been enabled can be measured. These are common configurations for those who frequently forget passwords.

- Password Metric 1: Enumerate user accounts that are not required to have a password
- Password Metric 2: Is auto-logon enabled (login occurs without entering a password)

3.2.2 Antivirus Metric

Windows readily provides information regarding antivirus configuration. This study verifies whether an antivirus product is registered with the operating system. A step further would be to query the state of software; however, product state information is vendor specific and would require a significant amount of logic to ascertain effectively.

- Antivirus Metric 1: Is antivirus software registered with the operating system as reported by Microsoft Windows

3.2.3 Privilege Metrics

Metrics chosen to evaluate privilege not only evaluate the access level of a user, but the number of users that are allowed access at all. This is an important factor with regard to shared computers that may have a multitude of individuals making configuration changes with differing degrees of authorization. In addition, any active automation or system users that run various processes on the machine are included in the total count as they may alter the security posture of the host in question.

- Privilege Metric 1: Total number of entries in the Windows "users" security group
- Privilege Metric 2: Total number of entries in the Windows "administrators" security group
- Privilege Metric 3: Number of currently logged on users (at time of metric measurement)

3.2.4 Software Updates Metrics

Whether or not software on a computer is up-to-date can be a difficult question to answer without asking every application installed for an update status. Any number of applications may be installed and each would report differently. "Software Updates" was the most commonly discussed category of recommendations in the reviewed publications and so deserves significant attention. As a result, finding easily and unobtrusively queried computer characteristics that provide insight into software update status requires an innovative approach.

A common side-effect of updating software, whether it be the operating system or installed application, is the need for a system reboot. As such, measuring the system up-time (i.e. the duration for which the host has been in an "on" or "sleep" state) will capture how much time has passed since reboot-requisite application updates have been applied. Querying system uptime is limited in its ability to measure software update cycles; it can indicate that important updates have not yet been installed, but does not directly measure software update status.

Apart from updates applied to software installed on the host is the update status of the operating system itself. The time since the last Windows update is easily ascertained through a simple OS query and was measured directly in this study.

Another metric that may be of use in indicating software update status is the number of applications that are altogether installed on the host. The logic being that a system with more software requires more frequent and thorough update cycles and may be prone to missing important updates. Again, this is an indirect measurement and is therefore limited in its efficacy.

Taken together, the number of applications installed and the time since the host was last reset may grant intuitive insight as to the status of application updates on a PC. However, should a correlation become evident between these metrics and real vulnerability, additional research would need to be conducted to determine if the cause of the correlation occurred as a direct result of out-of-date software.

- Software Updates Metric 1: Days since last Windows update
- Software Updates Metric 2: System uptime (days)
- Software Updates Metric 3: Number of installed applications

3.2.5 Firewalls Metrics

Direct measurement of firewall status is pretty straightforward in a Windows environment. As a result, two metrics are used to determine the effectiveness of firewalling on the system. Public and private firewall status in Windows is collected to determine whether it is simply on or off. Then, as a way to determine configuration status, the number of ports on which Windows is actively listening for incoming connections is measured.

- Firewalls Metric 1: Whether both private and public firewalling is enabled
- Firewalls Metric 2: Number of listening TCP ports

3.2.6 Media Devices Metric

As "media devices" have been identified as a potential cause for concern, a simple metric of their frequency of use may identify vulnerability. Therefore, querying the operating system for total number of removable drives was included as part of the study.

- Media Devices Metric 1: Total number of removable drives

3.3 Script Implementation

Figure 1 describes the PowerShell code that was used to gather the metrics described. New-line characters are added for legibility. Each metric was measured on a single script line, and metric numbering corresponds to the values listed in sections 3.2.1 through 3.2.6. As demonstrated in the code, the attributes gathered are reported as either an integer or Boolean value, which are stored in a variable to be output later in a comma separated values (CSV) file. This file is appended to for each host that is being measured. The specific methodology for gathering metrics utilizes commands that are compatible with PowerShell 2.0. Additional information regarding the functionality of the script is discussed later and the full code implementation can be found in Appendix A of this document.

Category	Metric	Code
Passwords	1	`$($(Get-WmiObject Win32_UserAccount -Filter LocalAccount=$true \| Where-Object {($_.Disabled -eq $false) -and ($_.PasswordRequired -eq $false)}).SID | Measure-Object -Line).Lines`
	2	`@{$true=$true;$false=$false}[$(Get-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon').AutoAdminLogon -eq 1]`
Antivirus	1	`@{$true=$true;$false=$false}[$($(Get-WmiObject -Namespace "root\SecurityCenter2" -Class AntiVirusProduct).DisplayName).Count -gt 0]`
Privilege	1	`$($([ADSI]"WinNT://Localhost/Users,Group").PSBase.Invoke("Members")).Count`
	2	`$($([ADSI]"WinNT://Localhost/Administrators,Group").PSBase.Invoke("Members")).Count`
	3	`$($(Get-WmiObject Win32_LoggedOnUser | Select-Object Antecedent -Unique).Antecedent | Measure-Object -Line).Lines`
Software Updates	1	`New-Timespan -Start $($(Get-WmiObject Win32_QuickFixEngineering | Sort-Object InstalledOn -Descending | Select-Object -First 1).InstalledOn) -End $(Get-Date) | Select-Object -ExpandProperty Days`
	2	`$((Get-Date) - ([DateTime]$(SystemInfo | Find /i "Boot Time").Replace('System Boot Time: ',''))).Days`
	3	`$(Get-ItemProperty HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\*).Count`
Firewalls	1	`@{$true=$true;$false=$false}[$(Netsh Advfirewall Show Private | Select-String 'State' | Out-String).Contains('ON') -and $(Netsh Advfirewall Show Public | Select-String 'State' | Out-String).Contains('ON')]`
	2	`[System.Net.NetworkInformation.IPGlobalProperties]::GetIPGlobalProperties().GetActiveTCPListeners().Count`
Media Devices	1	`$(Get-WmiObject Win32_LogicalDisk \| Where-Object {($_.DriveType -eq 2) -and ($_.Size -gt 0)} | Select-String DeviceID | Measure-Object -Line).Lines`

Figure 1: Code Implementation of Metrics Measurement

3.4 Vulnerability Scanning

In order to compare the results of measured computer characteristics against actual vulnerability data, Tenable's Nessus vulnerability scanning software was used. A vulnerability scan was performed on each host for which metrics were collected. Nessus was selected as it offers an in-depth analysis and is widely used in industry.

3.4.1 Vulnerability Scoring

Nessus vulnerability scores are based on the industry standard Common Vulnerability Scoring System (CVSS) as provided by a third party and the National Vulnerability Database (NVD) run by the National Institute of Standards and Technology (NIST). Plugins that are run during a scan currently vary between the CVSS v2 and CVSS v3 standards ("CVSS Scores in Tenable Plugins", 2017). As a result, the scoring provided by the scans is ubiquitous and not specific to Tenable software.

Nessus provides a plethora of information with each scan, but only the following measurements are included as part of the study:

- Total number of vulnerabilities ("low" through "critical")
- Number of "critical" and "high" vulnerabilities (CVSS Scores 7.0-10.0)

These two measurements juxtaposed the previously collected computer characteristics data will constitute the basis of analysis for identifying correlation. High and critical vulnerabilities are combined as a way to separate number and severity of vulnerabilities found. In addition, CVSS v2 does not differentiate between "high" and "critical" as CVSS v3 does, combining them removes any differences in Nessus plugins that may result in reporting inconsistency.

3.4.2 Nessus and Host Configuration

In order to obtain thorough results, authenticated Nessus scans were performed. That is, for each host an administrative username and password are provided for login. Without this feature, Nessus can only measure vulnerabilities that are exposed via open ports on the host. There are many vulnerabilities that may exist which can be exploited through a phishing or USB attack and credentialed scans allow us to measure those.

There are a number of configuration changes that must be made on each host such that Nessus can perform a complete scan remotely. These include:

- Running the Remote Registry Windows service
- Allowing Server Message Block (SMB) through the firewall
- Disabling User Access Control (UAC)
- Allowing Windows Management Instrumentation (WMI) through the firewall

The script that collects metrics from each host also configures it for Nessus scanning. The initial host settings are saved so that the host is restored to its original state once the scan is completed.

3.5 Data Collection Process

The following steps are taken each time a test of one or more hosts is performed:

- Administrative account is logged into at the host.
- USB flash drive with configuration and metrics collection script "Start-NessusPreparation.ps1" is connected to the host.
- Start-NessusPreparation.ps1 -NessusScanPrep is run at a PowerShell prompt from the directory of the flash drive indicating that we are both collecting host information as well as preparing it for Nessus scanning.

- The script outputs a configuration state file as well as a backup of firewall settings
- Script output is parsed manually, any errors are identified and resolved.
- Metrics collected are appended to “Results.csv”
- In the case of single-host testing, an ad-hoc network is created by attaching an ethernet cable between the host and the Nessus scanner PC. For multiple host scans, Start-NessusPreparation is run on each host before initiating a scan from the Nessus scanner over the network.
- A Nessus “Basic Network Scan” is configured with administrative credentials and appropriate IP address(es).
- Once the scan is finished, Start-NessusPreparation.ps1 -RestoreSettings is run on each host. The Windows firewall backup is restored and registry/service settings are reverted to their original states.

3.6 Data Analysis Process

After all data was collected Microsoft Excel was used to create scatter plots that demonstrated the relationship between computer characteristics and vulnerability. Two scatter plot graphs were created for each measured characteristic. The first related the characteristic to total vulnerability and the second related the characteristic to the combined number of high and critical vulnerabilities. Trendlines were plotted for each graph and correlations were calculated. Strong correlation coefficients were determined to be $r = 0.80$ or higher.

3.7 Hosts Identified for Study

At the completion of the study, a total of forty hosts were evaluated. These were primarily sourced from two environments whose administrators were willing to participate.

Details regarding subjects and location are not disclosed in order to keep vulnerability data anonymized.

3.7.1 University Cybersecurity Lab Machines

19 hosts were selected from a student lab in a university setting. These computers are utilized by students in several classes per semester. They are often used to perform research and complete assignments. Each class has unique requirements for the use of the lab machines putting them through broad range of tasks over long periods of time between maintenance activities. The computers are all domain joined and running Windows 10 Enterprise

3.7.2 Small Business Workstations

17 hosts were selected from a small manufacturing business. These computers are used by employees to perform tasks ranging from billing and accounting to 3D rendering and modeling. These machines are domain joined and primarily used by a single user per computer. They are running Windows 10 Professional.

3.7.3 Personal Computers

4 hosts were personal computers belonging to individuals who volunteered their computers for the study.

4 FINDINGS

This chapter summarizes the key findings from the analysis. The findings are organized into those that did not have sufficient data for correlational analysis (Section 4.1) and those that did have sufficient data (Section 4.2).

4.1 Non-Correlating Data

For several of the metrics collected there was insufficient variance in the results to be able to determine any correlation.

4.1.1 Auto Admin Logon Enabled

For every host in the study, none were configured to be able to login without the use of a password. No correlation was discovered.

4.1.2 Public and Private Firewalls Enabled

Every host in the study had both public and private firewalls enabled save for one only. No correlation was discovered.

4.1.3 Accounts Without Password Required

The majority of hosts had a password requirement for all users. The exception cases in which one or two accounts had no requirement, no correlation to vulnerability was discovered.

4.1.4 Number of External Drives

Each host in the study reported one removable storage device attached with one exception. No correlation was discovered.

4.1.5 Antivirus Installation

Every host tested had antivirus software installed. No correlation was discovered.

4.1.6 Number of Local Users and Administrators

The hosts in this study did not vary significantly in the number of local users or administrators. While it is possible that higher counts of users and administrators may still correlate to high vulnerability, the data here do not suggest any. This is primarily due to the use-cases for the specific machines that were used. A sample of computers that includes those that are used by many people would need to be tested to evaluate this criterion. Figure 2 shows the relationship between the number of local administrators and total number of vulnerabilities. The number of local users to total number of vulnerabilities relationship is much the same.

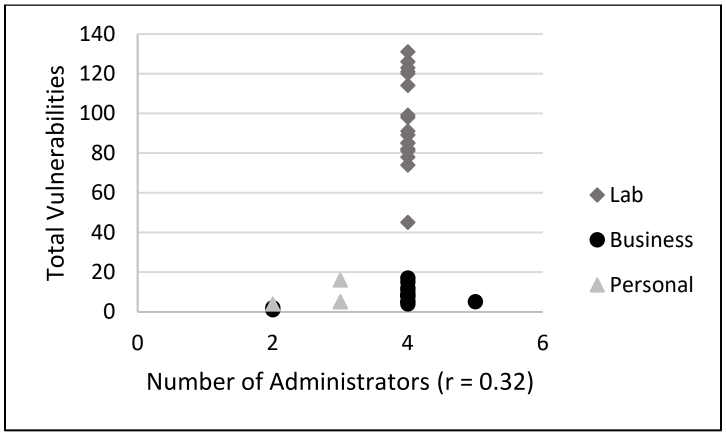

Figure 2: Total Vulnerabilities by Number of Local Administrators

4.2 Number of Programs Installed

Following are the results correlating the total number of installed programs with number and severity of vulnerabilities. As can be seen in Figure 4, a strong correlation (r = 0.90) was found between subsets of data when compared against total number of vulnerabilities. The correlation is moderate (r = 0.59) within the small business subset, but is reversed in the lab subject results (r = -0.68). Figure 5 demonstrates a similar trend for "High + Critical" (r = 0.88) vulnerability counts but with less demonstrable correlation in the small business subset

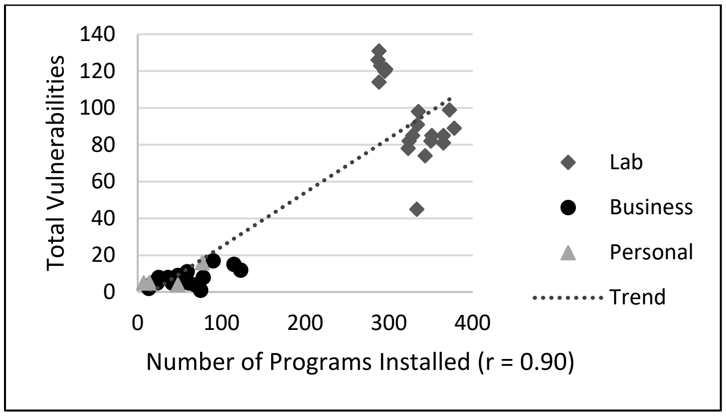

Figure 3: Total Vulnerabilities by Number of Programs Installed

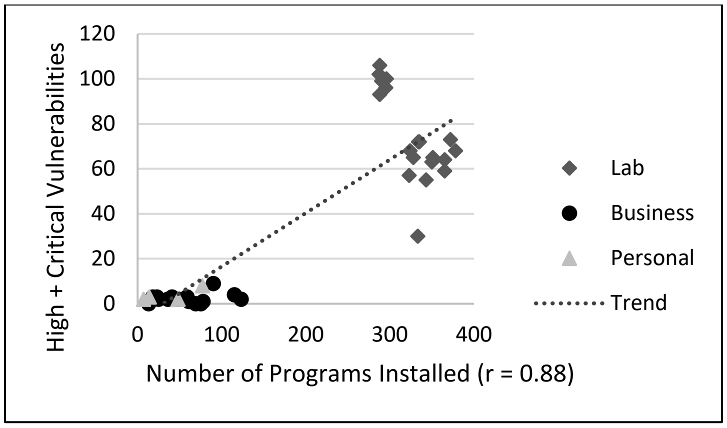

Figure 4: High and Critical Vulnerabilities by Number of Programs Installed

4.3 Number of Listening TCP Ports

Following are the results correlating the number of listening TCP ports with number and severity of vulnerabilities. In Figure 6 strong correlation (r = 0.82) was found when considering all hosts and total vulnerability counts. Correlation does not hold when considering the lab data subset alone (r = 0.03) and is moderate within the small business subset (r = 0.59). "High + Critical" counts against all hosts in Figure 7 reports a similar trend (r = 0.80) but with less demonstrable correlation in the small business subset.

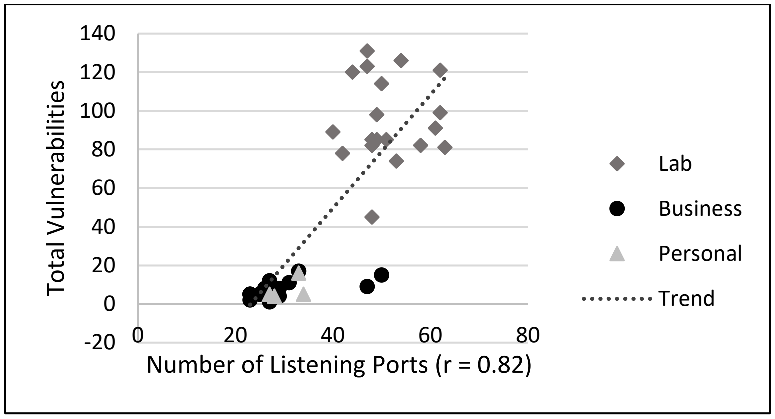

Figure 5: Total Vulnerabilities by Listening Ports

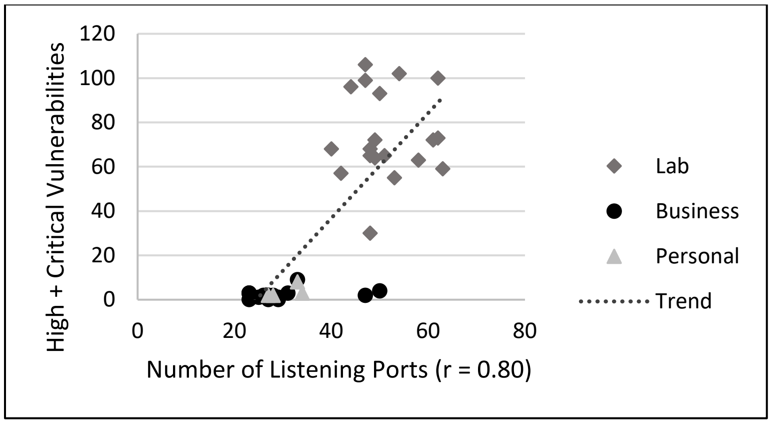

Figure 6: High and Critical Vulnerabilities by Listening Ports

4.4 Days Since Last Windows Update

Following are the results correlating the number of days since the last Windows update with number and severity of vulnerabilities. Figure 8 demonstrates strong correlation when considering all hosts (r = 0.93). Taken separately the lab (r = 0.19) and small business (r = 0.09) subsets show no meaningful correlation. Figure 9 shows similar trends for "High + Critical" vulnerabilities (r = 0.92).

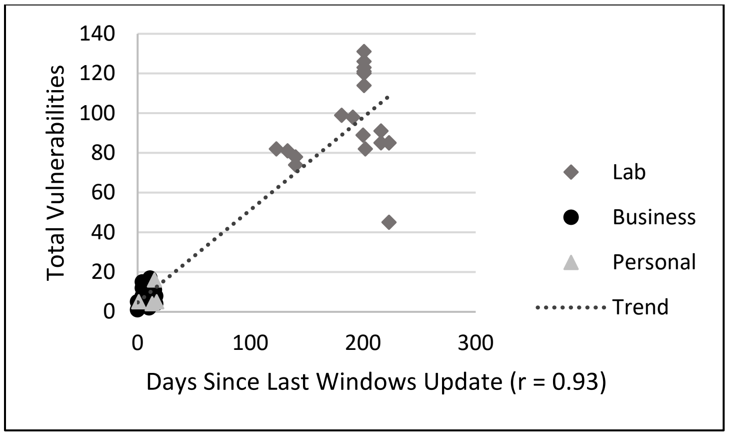

Figure 7: Total Vulnerabilities by Days Since Last Windows Update

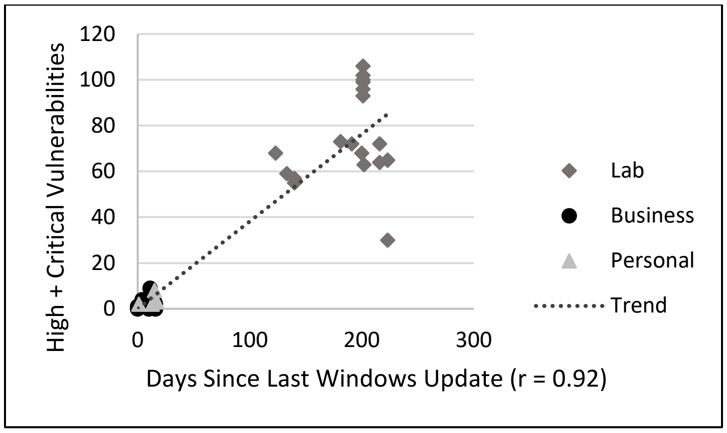

Figure 8: High and Critical Vulnerabilities by Days Since Last Windows Update

4.5 Number of Logged in Users

Following are the results correlating the number of concurrent logged in users with number and severity of vulnerabilities. This includes system users and service accounts. Figure 10 shows a moderate level of correlation when all hosts are considered (r = 0.68). Some correlation is maintained when the lab (r = 0.32) and business (r = 0.31) data sets are considered separately. Figure 11 shows similar trends for comparison with all hosts against "High + Critical" vulnerabilities (r = 0.67).

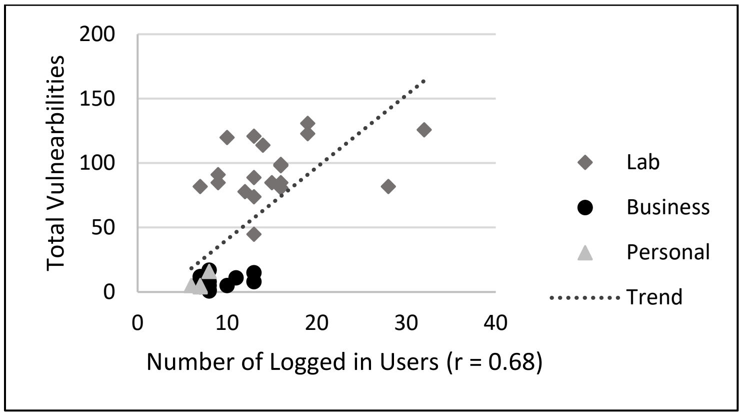

Figure 9: Total Vulnerabilities by Number of Logged in Users

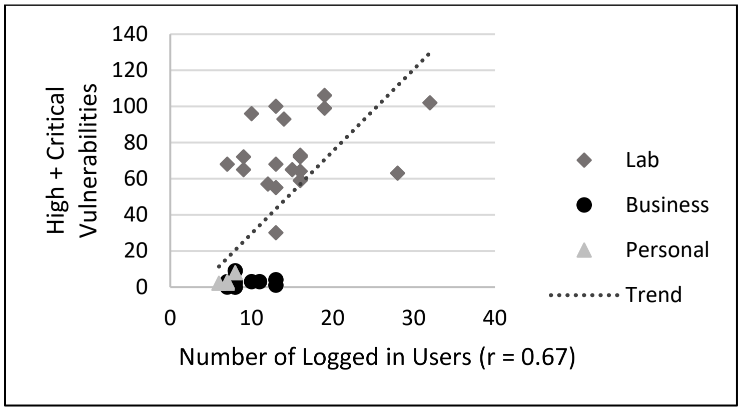

Figure 10: High and Critical Vulnerabilities by Number of Logged in

4.6 Days Since Last Restart

Following are the results correlating the number of days since the last Windows update with number and severity of vulnerabilities. Figure 12 shows that a strong correlation (r = 0.97) exists in total vulnerability count when all hosts are considered. Moderate correlation is maintained when considering the lab (r= 0.68) and small business (r = 0.57) subsets separately. Figure 13 demonstrates similar trends for "High + Critical" vulnerabilities (r = 0.96).

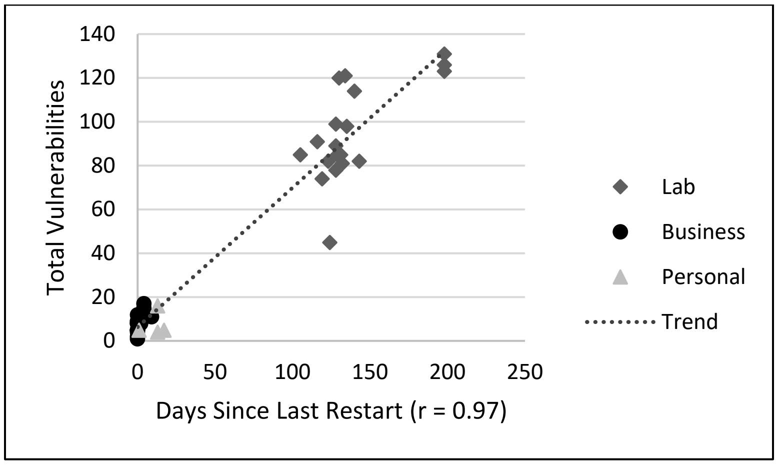

Figure 11: Total Vulnerabilities by Days Since Last Restart

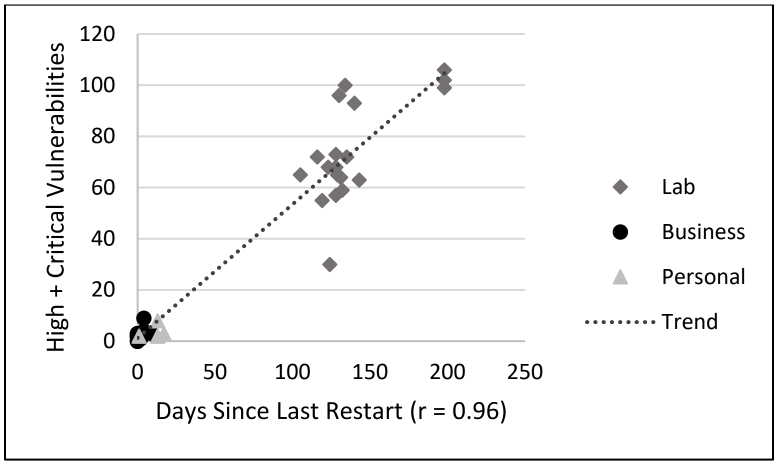

Figure 12: High + Critical Vulnerabilities by Days Since Last Restart

4.7 Summary of Results

Table 2 represents all corollary data as evaluated in the study.

Table 2: Summary of Correlation Results

	All Hosts (Total Vuln.)	All Hosts (High + Critical Vuln.)	Lab Data (Total Vuln.)	Business Data (Total Vuln.)
Password Not Required	N/A	N/A	N/A	N/A
Auto-Logon Enabled	N/A	N/A	N/A	N/A
Antivirus Installed	N/A	N/A	N/A	N/A
Number of Users	-0.26	-0.26	N/A	N/A
Number of Administrators	0.32	0.30	N/A	N/A
Logged On Users	0.68	0.67	0.32	0.31
Days Since Last Windows Update	0.93	0.92	0.19	0.09
Days Since Last Restart	0.97	0.96	0.68	0.57
Number of Installed Applications	0.90	0.88	-0.66	0.59
Number of Listening TCP Ports	0.82	0.80	0.03	0.59
Firewall Enabled	N/A	N/A	N/A	N/A
Number of Removable Devices	N/A	N/A	N/A	N/A

5 DISCUSSION AND CONCLUSIONS

This chapter discusses the results of the study and their significance, as well as the limitations and challenges faced. It concludes with an overview of how this work could be advanced in the future.

5.1 Differences in Hosts Studied

Before the study began, an effort was made to select hosts that would represent different use cases for personal computers whether they be utilized in the workplace, at home, school or whether the computers would be shared. The reason for this may be easily misunderstood. This research is not concerned with making sure that every use case is represented, but instead attempts to discover what correlations can be found regardless of how the technology is used. What is important is selecting hosts that supply a varying range of values for the metrics selected for measurement. The intention was to select sample sets where the ranges in metrics and vulnerability counts would be quite varied. It was impossible to know if these predictions were correct in this regard before taking measurements.

As can be seen from the data, there was a stark polarity between the two sets in several metrics. Referring to section 4.1 we also found that there was nearly complete uniformity in other metrics. This presents interesting challenges which we will discuss later on in the chapter.

The university lab the computers in question were not frequently maintained as students from many different classes relied on their consistent state. They may install specialized software, conduct long-term experiments, or even access them remotely if needed. This allows the students the flexibility they need in order to learn and explore effectively. With such frequent and varied use of lab computers by so many individuals, high counts of many metrics to be measured were expected. In contrast, the environment at the small business is tightly controlled by an IT administrator for the company. Any administrative activity on the machines is performed by him including the installation of software or the enabling of remote access. In this case low counts of many metrics to be measured were expected.

When taking the university lab and small business host subsets individually, correlation always diminishes and in one case even reverses. Where correlation holds when considering each subset on its own we find the best evidence that the correlation holds true generally. Interestingly, the metric which best predicts computer vulnerability was system uptime which was an indirect method for measuring computer update status. This correlation was even stronger than measuring operating system update status directly.

5.2 Total Vulnerability vs High Severity Vulnerability

This research attempts to understand computer characteristics that identify easily compromised hosts. I.e., discovering high-risk, vulnerable areas of the network without having to rely on in-depth vulnerability analysis. In this context, taking a total vulnerability count as given by any configuration or system state with which NIST assigns a CVSS score may not be satisfactory. There are a significant number of vulnerabilities that pose little danger. High counts of these, while potentially corollary to measured metrics, would not contribute to finding unprotected assets.

Despite the concern of low-severity vulnerabilities inhibiting the discovery of effective correlations, the data collected show little difference in results whether basing our evaluation on total number of vulnerabilities or only those with high or critical ratings. Though there is a consistent slight decline in correlation when considering only high and critical vulnerabilities, the differentiation is of little consequence.

5.3 Non-Correlating Metrics

In this study there were several metrics measured which did not show any significant correlation to vulnerability. However, given the sample size and general lack of variability in some of these measurements there is insufficient evidence to prove any correlation.

For example, every host in the study had antivirus installed. Despite having recorded a wide range in the number of vulnerabilities reported for our hosts, there is insufficient evidence to state that no correlation between antivirus installation and number or severity of vulnerabilities; other factors exist. All machines in the study were running Windows 10 which comes with Windows Defender software pre-installed. Older computers may have not had antivirus software installed were they part of the study. This would then present the possibility of finding correlation and delivered different results. Other metrics that suffer a similar challenge were “Accounts Without Password Required”, “Number of External Drives”, and “Public and Private Firewalls Enabled”

A couple metrics that are independent of factors such as operating system version showed some evidence of a lack of significant correlation. Little variance between hosts existed when measuring number of local users or administrators despite the range of vulnerability counts between them. Without this metric having other dependencies, there is some evidence that a correlation does not exist. However, a stronger, more appropriate argument would have been presented if a wider range of values had resulted from measurement of these metrics.

5.4 Correlating Metrics

Understanding that there is little difference in evaluating our metrics against total vulnerability count or against those rated as high or critical, the analysis will continue based only on total vulnerability counts. As a result of this study there is evidence to suggest that correlations exist between the following variables and number of vulnerabilities:

- Number of Programs Installed (r = 0.90)
- Number of Listening TCP Ports (r = 0.82)
- Days Since Last Windows Update (r = 0.93)
- Days Since Last Restart (r = 0.97)

In addition, there was evidence to show some correlation in:

- Number of Logged in Users (r = 0.68)

This now provides some basis from which to work as pertains to the development of tools that can use easily and unobtrusively queried computer characteristics to quickly identify potentially vulnerable areas on a network. This, of course, in the context of edge nodes running the Microsoft Windows operating system as far as the scope of this study is concerned.

5.5 Literature Review Conclusions

5.5.1 Firewalls

Windows Firewall has long been an included feature of the operating system which is enabled by default. It is no surprise, then, that our measurements indicate that each host in the study (with a single exception) has firewalling enabled. This is the recommendation that was given in several of the publications reviewed previously. However, there is no evidence that enabling this particular feature has any direct effect on the number of vulnerabilities that exist on a host. A second metric exists, "Number of TCP Listening Ports", that supplements firewall enablement as part of our literature evaluation. The availability of the host to accept external communication is determined by measuring the number of open ports. Many installed programs (safe, potentially unwanted, and malicious) may be opening up ports to accept network traffic and make outbound connections to other services. This grants some visibility into the configuration of the firewall as programs are frequently admitted access to and from the machine. In fact, pivoting on the data collected in this research, there is a strong positive correlation in the number of programs installed on a host and the number of listening TCP ports. It should be emphasized that this data does not prove causation, but may help deliver better recommendations to end users who are seeking to protect themselves from online threats.

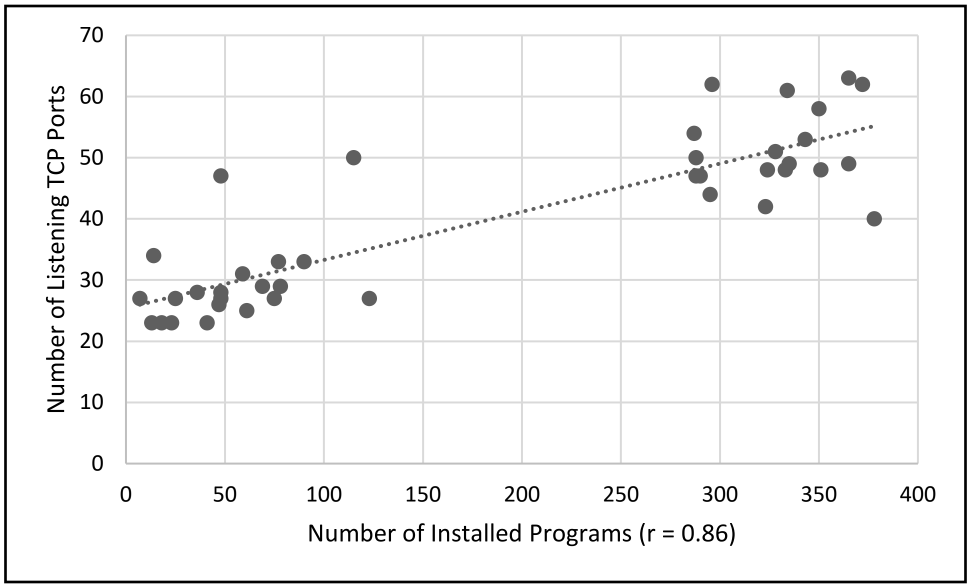

Figure 13: Number of TCP Listening Ports by Number of Installed Programs

Given the frequency with which firewall implementation is suggested, it may be wise to take this recommendation a step further. Firewalls should not only be enabled, but configured in such a way that limits the ability of programs to open holes that partially negate its effectiveness. This may be accomplished through the limiting of software installation in general which, as seen in figure 18, demonstrates how closely tied together these metrics are – in addition to their correlation with vulnerability.

5.5.2 Software Updates

Keeping software up to date was the most common recommendation made from the publications reviewed. This was the case as far as directly measuring the time since the last Windows update was concerned. In some cases, this was likely tied to the uptime of the system since many security updates cannot be applied until the system restarts. Again, this is also true of many installed applications. Even though many may update automatically, a program or system restart may be required for the update to take effect. Since both easily queried metrics of “Days Since Last Restart” and “Days Since Last Windows Update” correlate positively with vulnerability count, our previously held assumption that software updates are critical to system security holds and is supported by this study. This also contributes to the argument that forced computer restarts by the operating system may be an effective security measure.

5.5.3 Other Metrics

Password, media device, and privilege metric measurements did not yield any evidence that could support or challenge the recommendations that were presented in the literature review. The challenges faced in collecting meaningful data are discussed in the following section.

5.6 Study Limitations and Challenges

During the study there were a number of challenges faced that limited the usefulness of the collected data. These are discussed here.

5.6.1 Boolean Value Correlations

In order to collect data that could be evaluated against the number and severity of vulnerabilities, all of the metrics chosen are represented as either Boolean or integer values. It is more difficult to identify correlations with Boolean values because of their binary nature. More hosts would need to be included in the study to make up for the lack of possibilities in the value set. In addition, the particular measurements taken as Booleans were relatively uniform across all hosts. These included, “Antivirus Installed”, “Accounts Without Password Required”, “Auto Admin Logon Enabled”, and “Public and Private Firewalls Enabled”. These metrics were collected without consideration given to the default nature of the Windows Operating system and the likelihood that default settings would be adjusted. As a result, they did not add significant value to the study.

5.6.2 Diversity of Hosts Represented

Though the hosts tested vary in significant ways, one commonality between all of them is that all are running Windows 10. While Windows version is not a metric collected for evaluation, it is possible that it has influence on other metrics. This could have been an interesting point of discussion. A possible hypothesis might suggest that several correlation values would change with Windows version as it is easier to configure Windows update delays and avoid antivirus software in Windows 8.1 and earlier. Identifying system properties that may impact study results is a challenge; finding subjects that vary in every impactful system property would require a considerably larger sample size.

In addition, once the study was completed, the two major sample groups varied greatly in number of vulnerabilities as well as in many of the collected metrics for comparison. This did, however, leave a significant gap between the groups in which there were few representative hosts. While it is possible to extrapolate the data to fill in the gaps it would have been more effective to find a third sample group with middling metric values to determine if the number of vulnerabilities found is also an average between the two extremes. Though since the correlations found are so strong, even an attempt at finding a sample group with "middling metric values" demonstrates the strong interconnectedness of the data – many of these metrics not only correlate to increases in vulnerability counts, but to each other.

5.6.3 PowerShell 2.0

Since the study was designed before all test hosts were identified an effort was made to include any host running Windows 7 Service Pack 0 or higher. This means that the PowerShell script measuring metrics could only use functions available in version 2.0 or earlier which was released in 2009. Since then PowerShell functionality has increased tremendously and there are more effective ways of taking measurements than were implemented. Had it been known that all hosts tested would be running Windows 10, PowerShell functionality up to version 5.0 would have been included.

5.7 Future Work and Conclusion

To build on this research there are a number of questions to be answered and tools to be built that operate on this data. Within the scope of the Windows operating system, it must be understood whether the correlations represented in this research for personal computers and workstations are reflected in server platforms as well. New metrics can also be defined and the methods by which they are measured improved. Algorithms that calculate confidence of compromise should also be developed based on corollary data. This will require significantly more data so that a regression analysis can be performed to identify the best predictor of vulnerability, perhaps to the exclusion of all other metrics or as a combination.

Outside of the Windows operating system researchers should also examine many kinds of network nodes running a myriad of software. Similar techniques can be used to this end. With sufficient data, tooling can be developed which utilizes the technique of measuring easily and unobtrusively queried computer (or device) characteristics to predict vulnerability and therefore allow administrators to more quickly attend to actively securing a network. This study has established a methodology for examining these characteristics to discover correlation and demonstrated that computer characteristics indeed exist that can be used as vulnerability predictors. With additional research, refined measurements may provide powerful, actionable insight to organizations who intend to secure their networks immediately, before spending valuable time and money testing and implementing vulnerability assessment tools.

REFERENCES

Albanese, M., S. Jajodia, and S. Noel. 2012. “Time-Efficient and Cost-Effective Network Hardening Using Attack Graphs.” *Proceedings of the International Conference on Dependable Systems and Networks*. https://doi.org/10.1109/DSN.2012.6263942.

Banzhof, C., K. Cook, D. Helffrich, and R. Lawson. 2006. INVENTORY MANAGEMENT-BASED COMPUTER VUILNERABILITY RESOLUTION SYSTEM, issued 2006.

“CIS Controls.” n.d. *Security, Center for Internet*. Accessed December 6, 2018. https://www.cisecurity.org/controls/controlled-use-of-administrative-privileges/.

“CVSS Scores in Tenable Plugins.” 2017. Tenable Community. 2017. https://community.tenable.com/s/article/CVSS-Scores-in-Tenable-Plugins.

“Cyber Tips.” n.d. NJ Cybersecurity & Communications Integration Cell. Accessed August 23, 2018. https://www.cyber.nj.gov/cyber-tips.

“Cybersecurity for Small Business.” n.d. Federal Communications Commission. Accessed December 6, 2018. https://www.fcc.gov/general/cybersecurity-small-business.

Davis, G.. 2017. “10 Tips To Stay Safe Online.” McAfee Blog. 2017. https://securingtomorrow.mcafee.com/consumer/consumer-threat-notices/10-tips-stay-safe-online/.

Drapala, K.. 2013. “Top Ten: The Most Important Cyber Security Tips for Your Users.” Cisco Umbrella. 2013.

Eadicicco, L.. 2014. “12 Quick Internet Safety Tips That Will Save Your Digital Life From Getting Hacked.” Buisness Insider. 2014. https://www.businessinsider.com/how-to-prevent-cyber-attacks-2014-5.

“General Tips & Advice.” n.d. Department of Homeland Security. Accessed August 23, 2018. https://stopthinkconnect.org/tips-advice/general-tips-and-advice.

Hoffman, C.. 2017. “Basic Computer Security: How to Protect Yourself from Viruses, Hackers, and Thieves.” How-To-Geek. 2017. https://www.howtogeek.com/173478/10-important-computer-security-practices-you-should-follow/.

Kolbitsch, C., P. M. Comparetti, C. Kruegel, E. Kirda, X. Zhou, X. Wang, U C Santa Barbara, and Sophia Antipolis. 2009. “Effective and Efficient Malware Detection at the End Host.” *System* 4 (1): 351–366. https://doi.org/10.1093/mp/ssq045.

Meulen, R., and C. Pettey. 2017. “Gartner Forecasts Worldwide Security Spending Will Reach $96 Billion in 2018, Up 8 Percent from 2017.” Gartner Newsroom. 2017. https://www.gartner.com/newsroom/id/3836563.

Patel, A., M. Taghavi, K. Bakhtiyari, and J. Celestino Júnior. 2013. “An Intrusion Detection and Prevention System in Cloud Computing: A Systematic Review.” *Journal of Network and Computer Applications* 36 (1): 25–41. https://doi.org/10.1016/j.jnca.2012.08.007.

Sawik, T.. 2013. “Selection of Optimal Countermeasure Portfolio in IT Security Planning.” *Decision Support Systems* 55 (1): 156–64. https://doi.org/10.1016/j.dss.2013.01.001.

Shacklett, M.. 2018. “10 Ways to Develop Cybersecurity Policies and Best Practices.” ZDNet Special Feature. 2018. https://www.zdnet.com/article/10-ways-to-develop-cybersecurity-policies-and-best-practices/.

Soderberg, J., and B. Kachachi. 2012. COMBINING ASSESSMENT MODELS AND CLIENT TARGETING TO DENTIFY NETWORKSECURITY VUILNERABILITIES, issued 2012. https://doi.org/10.1016/j.(73).

“Top 10 Secure Computing Tips.” n.d. UC Berkeley. Accessed December 6, 2018. https://security.berkeley.edu/resources/best-practices-how-to-articles/top-10-secure-computing-tips.

“Top Ten Safe Computing Tips.” n.d. MIT IST. https://ist.mit.edu/security/tips.

Viduto, V., C. Maple, W. Huang, and D. López-Peréz. 2012. “A Novel Risk Assessment and Optimisation Model for a Multi-Objective Network Security Countermeasure Selection Problem.” *Decision Support Systems* 53 (3): 599–610. https://doi.org/10.1016/j.dss.2012.04.001.

WaterISAC. 2015. “10 Basic Cybersecurity Measures: Best Practices To Reduce Exploitable Weaknesses And Attacks.” https://ics-cert.us-cert.gov/sites/default/files/documents/10_Basic_Cybersecurity_Measures-WaterISAC_June2015_S508C.pdf.

WP BrandStudio. 2017. “The 5 Cybersecurity Practices Every Employee Should Follow.” The Washington Post. 2017. https://www.washingtonpost.com/sf/brand-connect/wp/2017/11/07/att/the-5-cybersecurity-practices-every-employee-should-follow/?noredirect=on&utm_term=.fe453997f427.

Zaharia, A.. 2016. “50+ Internet Security Tips & Tricks from Top Experts.” Heimdal Security. 2016. https://heimdalsecurity.com/blog/security-experts-roundup/.

Appendix A. Full Code Listing

```
[cmdletbinding(DefaultParameterSetName="HostInfo")]
param (

   # - A description is required if collecting "host info" i.e. "easily and unobtrusively queried
computer characteristics".
   # - Modifying Host configuration in preparation for Nessus Scanning requires a description
unless not collecting
   #    host info with -SkipHostInfo
   # - Restoring original host settings does not collect host info and does not require a
description
   # - By default the script collects host info without making any configuration modifications

   [Parameter(ParameterSetName="HostInfo", Mandatory=$false)]
   [Switch]$HostInfo,

   [Parameter(ParameterSetName="HostInfo", Mandatory=$true)]
   [String]$Description,

   [Parameter(ParameterSetName="RestoreSettings", Mandatory=$true)]
   [switch]$RestoreSettings,

   [Parameter(ParameterSetName="NessusScanPrep", Mandatory=$false)]
   [Parameter(ParameterSetName="HostInfo", Mandatory=$false)]
   [switch]$NessusScanPrep,
   [switch]$SkipHostInfo
)

Start-Transcript -path "$($env:computername)Transcript.txt" -append

function Get-HostInfo {

   Write-Host "Collecting Mechine Information `n"

   $PowerShellVersion = $PSVersionTable.WSManStackVersion.Major
   $IPAddress = $($(Get-WmiObject Win32_NetworkAdapterConfiguration| Where-Object
{($_.Ipaddress.Length -gt 1) -and $null -ne $_.DefaultIPGateway}).IPAddress -Split '\n')[0]

   # - Collect each of the 12 easily queriable computer attributes

   $WindowsVersion = $(Get-WmiObject Win32_OperatingSystem).Caption

   $LocalAdministratorsCount =
$($([ADSI]"WinNT://Localhost/Administrators,Group").PSBase.Invoke("Members")).Count

   $LocalUsersCount =
$($([ADSI]"WinNT://Localhost/Users,Group").PSBase.Invoke("Members")).Count

   $AccountsWithoutPasswordRequired = $($(Get-WmiObject Win32_UserAccount -Filter
LocalAccount=$true | Where-Object {($_.Disabled -eq $false) -and ($_.PasswordRequired -eq
$false)}).SID | Measure-Object -Line).Lines

   $AutoAdminLogonEnabled = @{$true=$true;$false=$false}[$(Get-ItemProperty -Path
'HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon').AutoAdminLogon -
eq 1]

   $DaysSinceLastWinUpdate = New-Timespan -Start $($(Get-WmiObject
Win32_QuickFixEngineering | Sort-Object InstalledOn -Descending | Select-Object -First
1).InstalledOn) -End $(Get-Date) | Select-Object -ExpandProperty Days

   $DaysSinceLastRestart = $((Get-Date) - ([DateTime]$(SystemInfo | Find /i "Boot
Time").Replace('System Boot Time:          ',''))).Days

   $AntiVirusInstalled = @{$true=$true;$false=$false}[$($(Get-WmiObject -Namespace
"root\SecurityCenter2" -Class AntiVirusProduct).DisplayName).Count -gt 0]

   $PublicAndPrivateFirewallsEnabled = @{$true=$true;$false=$false}[$(Netsh Advfirewall
Show Private | Select-String 'State' | Out-String).Contains('ON') -and $(Netsh Advfirewall Show
Public | Select-String 'State' | Out-String).Contains('ON')]

   $NumberOfProgramsInstalled = $(Get-ItemProperty
HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\*).Count

   $NumberOfListeningTCPPorts =
[System.Net.NetworkInformation.IPGlobalProperties]::GetIPGlobalProperties().GetActiveTCPL
isteners().Count
```

```
    $NumberOfLoggedInUsers = $($(Get-WmiObject Win32_LoggedOnUser | Select-Object
Antecedent -Unique).Antecedent | Measure-Object -Line).Lines

    $NumberOfExternalDrives = $(Get-WmiObject Win32_LogicalDisk | Where-Object
{($_.DriveType -eq 2) -and ($_.Size -gt 0)} | Select-String DeviceID | Measure-Object -
Line).Lines

    $Data = @{'WindowsVersion'=$WindowsVersion;
    'IPAddress'=$IPAddress;
    'Description'=$Description;
    'LocalAdministratorsCount'=$LocalAdministratorsCount;
    'LocalUsersCount'=$LocalUsersCount;
    'AccountsWithoutPasswordRequired'=$AccountsWithoutPasswordRequired;
    'AutoAdminLogonEnabled'=$AutoAdminLogonEnabled;
    'DaysSinceLastWindowsUpdate'=$DaysSinceLastWinUpdate;
    'DaysSinceLastRestart'=$DaysSinceLastRestart;
    'AntivirusInstalled'=$AntiVirusInstalled;
    'PublicAndPrivateFirewallsEnabled'=$PublicAndPrivateFirewallsEnabled;
    'NumberOfProgramsInstalled'=$NumberOfProgramsInstalled;
    'NumberOfListeningTCPPorts'=$NumberOfListeningTCPPorts;
    'NumberOfLoggedInUsers'=$NumberOfLoggedInUsers;
    'NumberOfExternalDrives'=$NumberOfExternalDrives}

    $Results = New-Object -TypeName PSObject -Property $Data

    $Results | Select-Object *


    if ($PowerShellVersion -gt 2) {
        if(Test-Path .\Results.csv) {
            $Results | Export-CSV "Results.csv" -Append -Force
        }
        else {
            $Results | Export-CSV "Results.csv" -Force
        }
    }
    else {
        $Results | Export-CSV "$Description.csv"
    }
}

# - Administrator rights check
```

```
if (!([Security.Principal.WindowsPrincipal]
[Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuilt
InRole] "Administrator")) {
   Write-Warning "Script is not running with Administrator rights`nRe-run this script as an
Administrator"
   Break
}

if($NessusScanPrep) {

   Write-Host "Preparing Host for scanning `n"

   # - Check host configuration and set variables
   If($null -ne $(Get-ItemProperty -Path
"HKLM:\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile\Services\FileAndPr
int" -ErrorAction SilentlyContinue | Select-Object -ExpandProperty "Enabled")) {
      $FileandPrintRemoteAvailable = $($(REG query
"HKLM\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile\Services\FileAndPri
nt" /v Enabled) | Out-String).contains("0x1")
   }
   else {
      $FileandPrintRemoteAvailable = $false
   }

   If($null -ne $(Get-ItemProperty -Path
"HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" | Select-Object -
ExpandProperty "ConsentPromptBehaviorAdmin" -ErrorAction SilentlyContinue)) {
      $UACDisabled = $($(REG query
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v
ConsentPromptBehaviorAdmin) | Out-String).contains("0x0")
   }
   else {
      $UACDisabled = $false
      $UACString = REG query
"HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v
ConsentPromptBehaviorAdmin | Out-String
      switch ($UACString) {
         {$_.Contains('0x1')}{$UACStatus = 1}
         {$_.Contains('0x2')}{$UACStatus = 2}
         {$_.Contains('0x3')}{$UACStatus = 3}
         {$_.Contains('0x4')}{$UACStatus = 4}
         {$_.Contains('0x4')}{$UACStatus = 5}
         default {Write-Host "Problem reading UAC status in registry: `n$UACString"}
```

```
        }
    }

    If($null -ne $(Get-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\services\RemoteRegistry" | Select-Object -
ExpandProperty "Start" -ErrorAction SilentlyContinue)) {
        $RemoteRegistryInfo = REG query
"HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start | Out-String
    }
    else {
        Write-Host "Remote Registry Service Not Installed - Registry dependent Nessus plugins
will not run"
        $RemoteRegistryInfo = "Remote Registry Service Not Installed"
    }

    # - Collect host information unless otherwise specified
    if(!$SkipHostInfo) {
        Get-HostInfo
    }

    # - Backup host configuration
    Write-host "Saving configuration state to file"
    "$FileandPrintRemoteAvailable, $UACDisabled, $RemoteRegistryInfo, $UACStatus" | Out-
File -FilePath "SettingsBackup$env:computername.txt" -Force

    Write-Host "Exporting backup Firewall rules"
    netsh advfirewall export "FirewallPolicyBackup$env:computername.wfw"

    # - Make configuration changes
    if($FileanPrintRemoteAvailable) {
        Write-Host "File and printer remote access enabled, skipping registry modification"
    }
    else {
        Write-Host "File and printer remote access not enabled, modifying registry"
        REG ADD
"HKLM\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile\Services\FileAndPri
nt" /v Enabled /t REG_DWORD /d 1 /f
    }

    if(!$UACDisabled) {
        Write-Host "UAC is enabled, disabling"
        REG ADD "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v
ConsentPromptBehaviorAdmin /t REG_DWORD /d 0 /f
```

```
   }
   else {
      Write-Host "UAC is disabled, not making modifications"
   }

   Write-Host "Turning on remote registry service"
   REG add "HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start /t
REG_DWORD /d 2 /f
   net start RemoteRegistry

   Write-Host "Opening Firewall for Nessus SMB and WMI access"
   netsh advfirewall firewall set rule group="File and Printer Sharing" new enable=Yes
   netsh advfirewall firewall set rule group="windows management instrumentation (wmi)" new
enable=yes

   # - Look for Ad-hoc network to simplify Nessus scan configuration
   $IPAddressForNessusUse = $($(Get-WmiObject Win32_NetworkAdapterConfiguration|
Where-Object {($_.Ipaddress.Length -gt 1) -and ($_.Ipaddress | Out-
String).StartsWith("169")}).IPAddress -Split '\n')[0]

   if($null -ne $IPAddressForNessusUse) {
      Write-Host "`n Configure Nessus with the following IP Address: $IPAddressForNessusUse
`n" -ForegroundColor Yellow
   }
   else {
      Write-Host "`n Ad-hoc network not detected for Nessus scan - Attach network cable or use
LAN" -ForegroundColor Yellow
   }
}

elseif($RestoreSettings)
{

   # - Collect information from Settings backups
   Write-Host "Retrieving Previous settings from file"

   if($(Test-Path "SettingsBackup$env:computername.txt") -and $(Test-Path
"FirewallPolicyBackup$env:computername.wfw")) {
      $SettingsRetrieval = $(Get-Content "SettingsBackup$env:computername.txt" | Out-String) -
split ','
   }
   else {
      Write-Warning "Unable to locate SettingsBackup$env:computername.txt or
FirewallPolicyBackup$env:computername.wfw, run restore from the directory where files are
located"
      Break
   }

   $FileandPrintRemoteAvailable = $SettingsRetrieval[0]
   $UACDisabled = $SettingsRetrieval[1]
   $RemoteRegistryInfo = $SettingsRetrieval[2]
   if(!$UACDisabled){
      $UACStatus = $SettingsRetrieval[3]
   }

   # - Restore configuration changes
   Write-Host "Importing backup Firewall rules"
   netsh advfirewall import "FirewallPolicyBackup$env:computername.wfw"

   Write-Host "Restoring Remote Registry Service settings"
   switch ($RemoteRegistryInfo) {
      {$_.Contains('0x1')}{REG add
"HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start /t REG_DWORD /d 1
/f}
      {$_.Contains('0x2')}{REG add
"HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start /t REG_DWORD /d 2
/f}
      {$_.Contains('0x3')}{REG add
"HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start /t REG_DWORD /d 3
/f}
      {$_.Contains('0x4')}{REG add
"HKLM\SYSTEM\CurrentControlSet\services\RemoteRegistry" /v Start /t REG_DWORD /d 4
/f}
      default {Write-Host "No previous startup configuration for the Remote Registry Service
detected, will not modify. `n $RemoteRegistryInfo"}
   }

   if(!$UACDisabled) {
      Write-Host "Reenabling UAC settings"
      REG ADD "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" /v
ConsentPromptBehaviorAdmin /t REG_DWORD /d $UACStatus /f
   }

   if($FileandPrintRemoteAvailable) {
      Write-Host "Redisabling Remote File and print availablitity"
      REG ADD
"HKLM\SOFTWARE\Policies\Microsoft\WindowsFirewall\DomainProfile\Services\FileAndPri
nt" /v Enabled /t REG_DWORD /d 0 /f
   }

   $DeleteBackups = Read-Host -Prompt "Would you like to delete
SettingsBackup$env:computername.txt and FirewallPolicyBackup$env:computername.wfw?
yes/no default:no"

   if($DeleteBackups -eq "yes") {
      Remove-Item -Path "SettingsBackup$env:computername.txt" -Force
      Remove-Item -Path "FirewallPolicyBackup$env:computername.wfw" -Force
      Write-Host "Backup files deleted"
   }
   else {
      Write-Host "Backup files not deleted"
   }
}

else {
   Get-HostInfo
}

Stop-Transcript
```