KioskKit:

Project Report

Jonathan Foots

University of South Alabama

Executive Summary

As an Information Security Intern at the University of South Alabama, part of the job is to secure University machines that handle Personally Identifiable Information (PII) such as credit card numbers, bank account numbers, and social security numbers. This is important to the University in order to maintain compliance with the Payment Card Industry Data Security Standard (PCI DSS) and "lowering the risk profile of the University's electronic information by implementing industry best practices to protect the confidentiality, integrity, and availability of student, faculty, and staff information" (Information Security). The proposed KioskKit is an automation solution to configure any Windows 10 machine to adhere to the industry data security standards while requiring minimal input to apply the KioskKit. It will speed up the configuration process while eliminating potential missed steps from performing the configuration manually. With this KioskKit, the Information Security Office (InfoSec) as well as Academic Computing (ACAD) will be able to quickly deploy compliant machines into the University environment.

Problem Identification

For the University to comply with the Payment Card Industry Data Security Standard (PCI DSS), new Personal Computer (PC) setup must include hardening before it can be delivered to a department to begin handling PII. Hardening a PC is the process of restricting administrative tools and elevation of privileges of the standard user's account, called a Kiosk account. Hardening is a lengthy process if performed manually and typically takes between an hour to two hours depending on the degree of the hardening necessary for the machine. A similar solution would be to create a disk image of an already hardened machine and copy it to a new machine. However, the imaging solution requires both machines to have similar hardware drivers which is not always the case. Additionally, the imaging solution has reportedly caused issues with programs that rely on a unique machine identifier or a unique program ID that is generated on installation, such as Cisco AMP, Symantec, and Qualys. KioskKit will avoid these issues altogether by configuring the machine without copying everything as a disk image would.

Stakeholders for the KioskKit project include InfoSec, ACAD, interns working for InfoSec or ACAD, myself, School of Computing, University of South Alabama, the professor, and Kiosk users. InfoSec is responsible for creating the hardening standard for PC setup; however, ACAD is responsible for performing the PC setup. Interns for InfoSec or ACAD are the ones who will apply the KioskKit to the machines. I am responsible for creating the KioskKit. The School of Computing holds the Senior Demonstration Class. University of South

Alabama offers the Information Technology degree. The professor teaches the class and grades the project. Lastly, kiosk users are the ones who will use the machine after it has been deployed in the University environment.

Primarily, it is the interns who will reap the benefits of the KioskKit as they will not have to spend as much time configuring the machine manually. Additionally, InfoSec will be able to prove that the machines are compliant as performing it manually has the possibility of missed steps.

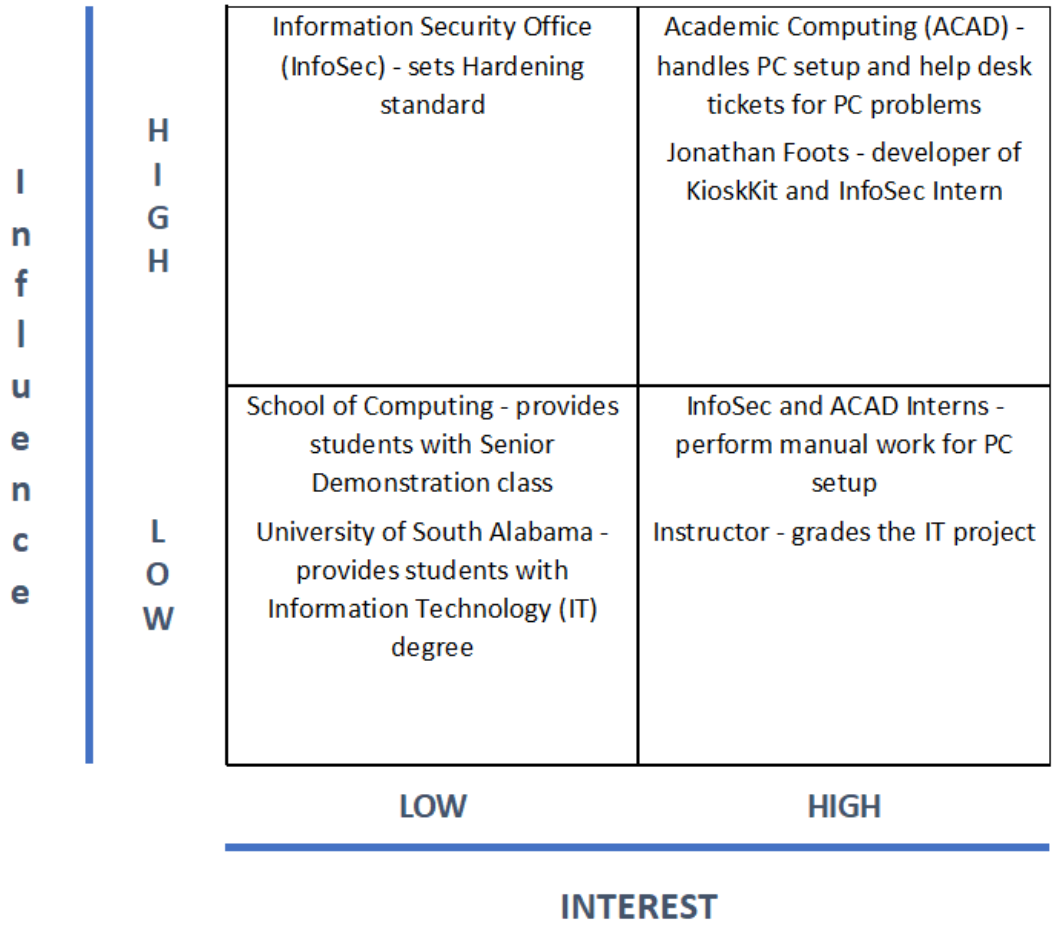

Figure 1

Referring to Figure 1 above, InfoSec has High influence and Low interest in the KioskKit because they are responsible for creating the hardening standard but do not actually perform the configuration. Next, ACAD has High influence

and High interest because they perform the configuration and handle related tickets for new PC setups. Next, interns have Low influence and High interest because the KioskKit automates the configuration process, but they do not have control over the hardening standard. Lastly, kiosk users have Low influence and interest as they only interact with the machine after it has been configured.

Evaluation of technology requirements

In order to complete the KioskKit, the following hardware and software will be needed: laptop, Virtual Box, Windows 10 LTSC ISO, and CIS-CAT Pro. The software will be installed on the laptop prior to development and testing. Testing will require a virtual environment to test configurations on as well as the ability to restore the environment to its previous configuration.

Windows 10 Long-Term Servicing Channel (LTSC) was selected as the operating system for the virtual machine (VM) because the "functionality and features don't change over time" (Wilcox, 2005, p. 3). Windows 10 LTSC only receives an update in the spring and fall on a semi-annual cycle; therefore, it is a more stable version of Windows 10 (Wilcox, 2005). Additionally, Windows 10 LTSC has no manufacturer software or unnecessary Windows store apps installed (Wilcox, 2005). This operating system is used specifically as part of the hardening process for PCI DSS compliant machines.

For demonstration purposes, Virtual Box (VBox) was selected to create a virtual environment because it was open source and free to use. VBox will be used for the creation of the VM using the Windows 10 LTSC ISO file to install a new operating system (OS) on it. Additionally, VBox will be used to configure the VM's specifications, such as system memory, number of CPUs, video memory, and disk space. The VM will be configured to ensure that it has enough hardware resources to run the OS without any issues.

KioskKit is intended to be applied from a flash drive or other storage medium to a fresh install of Windows 10 immediately after the administrative user logs for the first time.

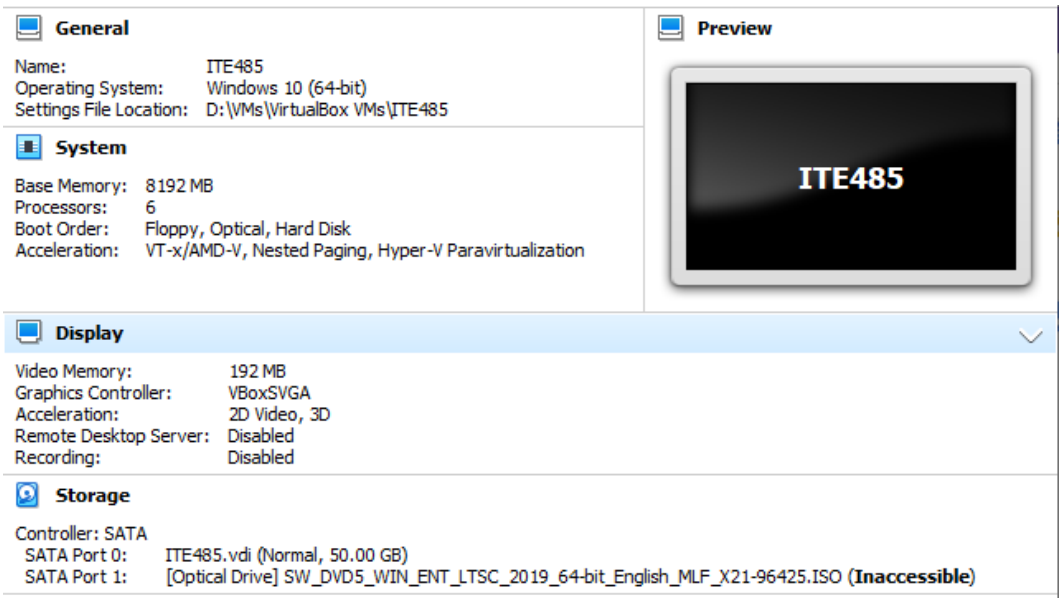

Figure 2

Referring to Figure 2 above, a new VM was created with 8192MB of system memory, 6 CPUs, 192MB of video memory, and 50GB of disk space. Next, the Windows 10 LTSC ISO was used to install the operating system on it. A device filter was added to allow the use of the USB with the KioskKit within the virtual environment. After, the VM was turned on, logged in to test USB connectivity. Lastly, I created a snapshot of the current state so that it can be restored after each test.

In order to apply the industry standard group policy settings, CIS-CAT Pro will be used as a remediation kit to apply group policy settings to the machine. CIS-CAT Pro is applied using a local group policy object utility (Margosis, 2016, p. 1). Applying through this tool is a requirement for compliance.

Prototype design specification

Referring to figure 3 below, deployment will follow these steps: installing the OS, logging in, retrieving the KioskKit files, running the script, waiting for it to finish, and restarting the computer.

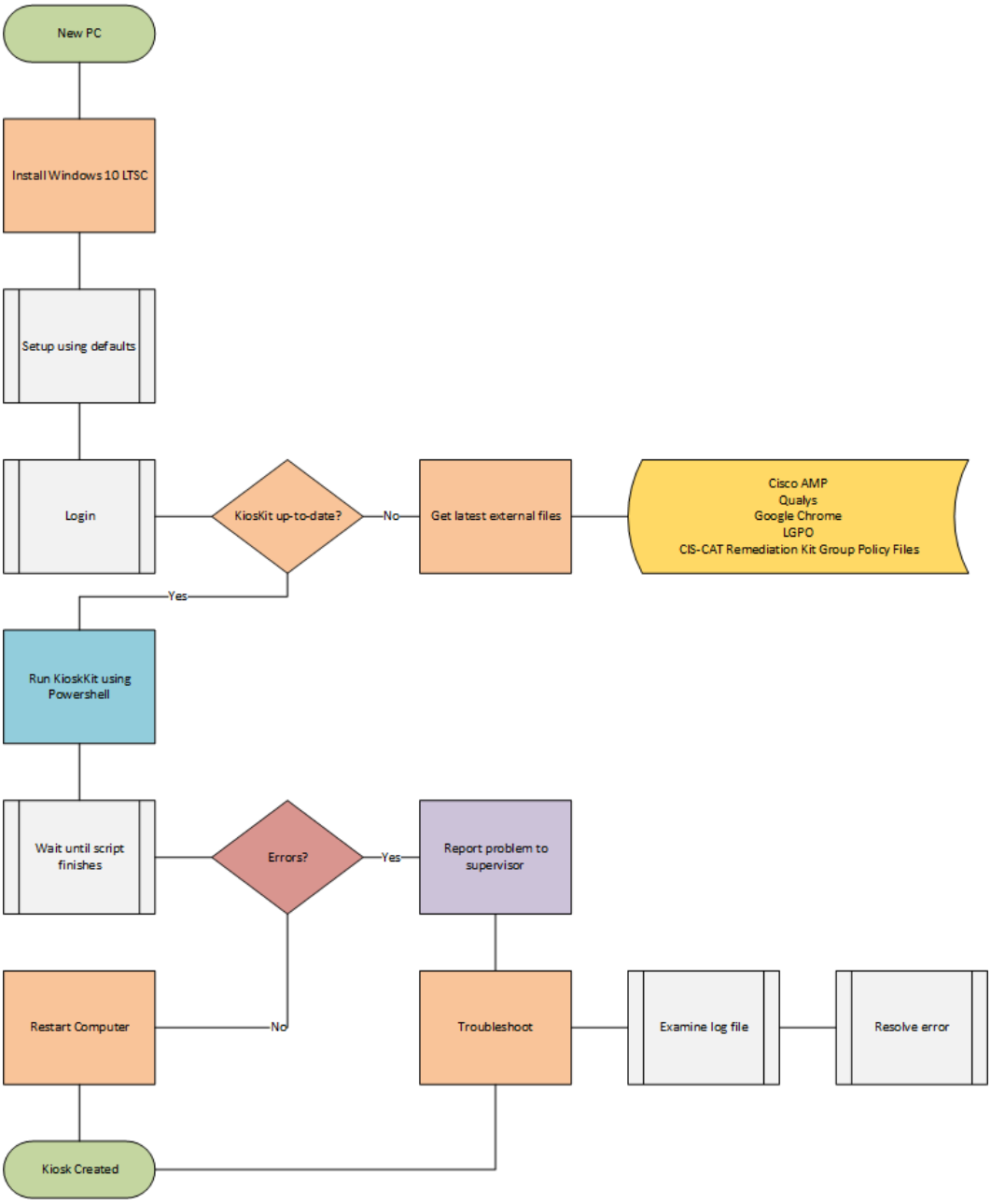

Figure 3

Also, the alternative flow in case of an error is described as: reporting to a supervisor, examining the error, and resolving the error.

For demonstration purposes, the laptop must have the hardware specifications to run a Windows 10 LTSC VM with at least minimum specifications needed for the OS while maintaining native OS of the laptop. Remember that the VM was created with 8192MB of system memory, 6 CPUs, 192MB of video memory, and 50GB of disk space. Therefore, the laptop must have at least double these hardware specifications. Additionally, VM must also be configured for a USB filter to permit the use of a USB with the KioskKit. Lastly, deploying the KioskKit from a machine that is not a VM will begin after logging into the PC and applying the KioskKit. KioskKit will apply the same way with a VM as it would a PC regardless if it was a fresh install or PC with prior usage.

During testing, the VM will restore the fresh install snapshot to revert changes. This is performed when powering off the VM before doing another round of testing. If this step is not performed, the KioskKit may produce unexpected results or errors. Additionally, this means that a KioskKit should not be applied more than once on the same physical machine.

Referring to figure 4 below, KioskKit will perform these steps: open powershell, access KioskKit files, execute KioskKitCore, prompt the user to press enter to proceed when ready, and recording results when finished. KioskKit breaks down into several scripts with KioskKitCore as the controlling script to execute them in sequence. KioskKit has the following sub-scripts: GroupPolicy, Registry, LocalSecurityPolicy, TaskScheduler, UserSettings, InstallsUpdates.

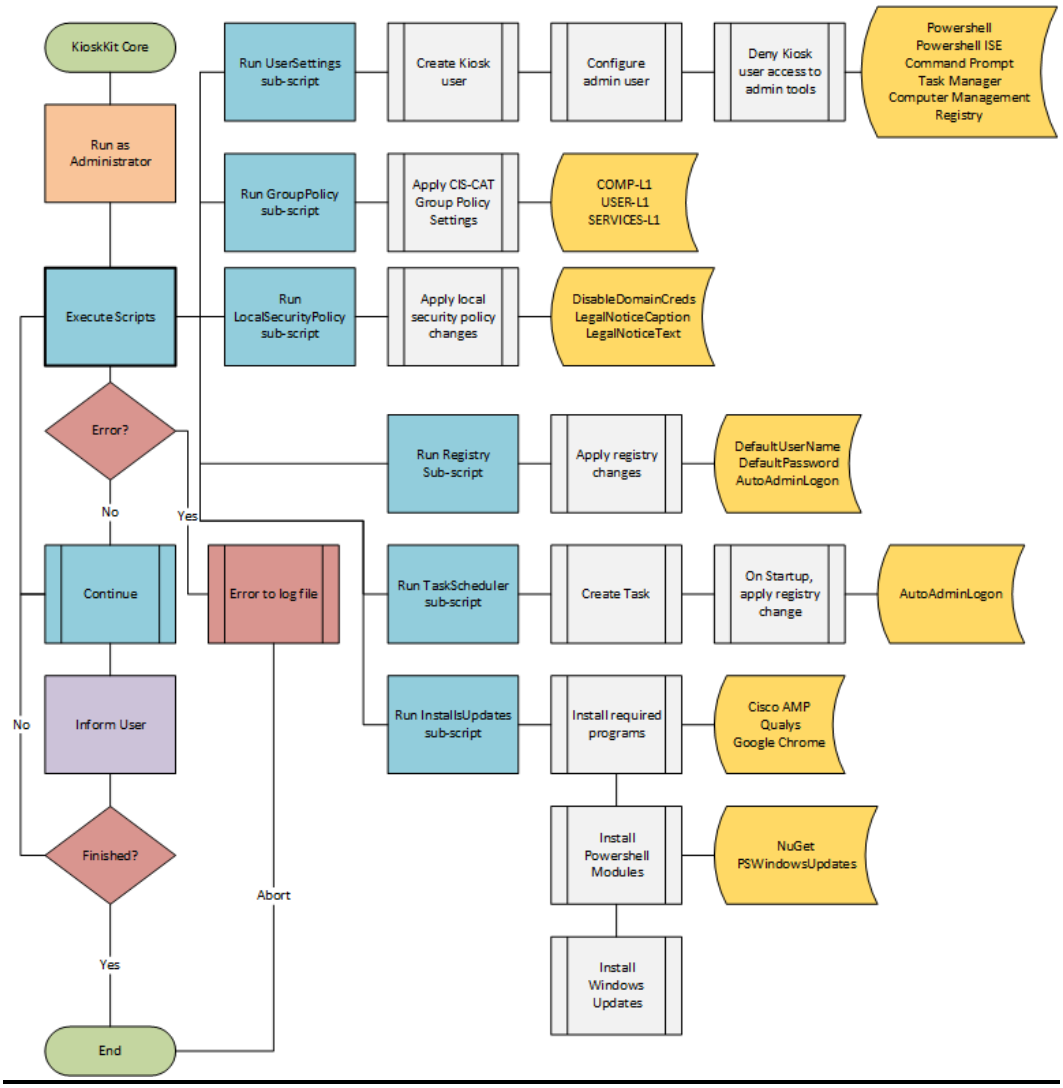

Figure 4

First, the UserSettings sub-script will create the Kiosk user account and deny access to administrative tools. Next, the GroupPolicy sub-script is responsible for applying the CIS-CAT remediation kit which will apply the industry standard level 1 group policy settings. Since this standard is required for compliance, the KioskKit will need to be updated with the latest remediation kit files on a semi-annual basis according to the Windows 10 fall and spring update cycle. Next, the LocalSecurityPolicy sub-script will configure the local security policy to allow tasks to be scheduled and set the default University acceptable-usage message at

the login screen. Next, the Registry sub-script is responsible for configuring the auto login of the standard user account. This will change the registry to set the default user and auto logon to the Kiosk account. Next, the TaskScheduler sub-script will create a scheduled task that will re-apply the auto login registry setting after the admin user account is used to ensure it continues to work. Lastly, InstallsUpdates sub-script will install Chrome, Qualys, and Cisco AMP, then the KioskKit will finish with Windows Update. After Windows Update, the PC will restart and will be ready to be deployed into the University environment.

Prototype implementation and evaluation

Referring to figure 5 below, PC setup performed by an intern will follow this scenario: logging into the computer, executing KioskKitCore, the sub-script will apply its settings, script will prompt the user to advance to the next script, KioskKit will finish with Windows Updates, user will restart the computer, and user will verify KioskKit has been applied. If an error has occurred, the error will be recorded in an error log that can then be reviewed and the script can be re-run. However, a fresh install of the OS may be required depending on the error. Alternatively, the specific sub-script that encountered the error may be run separately instead of the KioskKitCore script.

Use Case Name:	Apply KioskKit
Scope:	Turn a freshly installed Windows PC into a hardened Kiosk.
Level:	Low Influence/High Interest
Primary Actor:	Information Security Intern / ACAD Intern
Stakeholders:	InfoSec and ACAD who employ interns to do the work their offices are responsible for, Jonathan Foots who is an InfoSec intern and developer of KioskKit.
Preconditions:	Logged in with administrative privileges on a PC with a freshly installed Windows 10 LTSC operating system.
Postconditions:	Hardened Kiosk ready to be boxed and shipped to its destination.
Main Success Scenario (aka Flow of Events)	
Actor Action	**System Response**
1. Access KioskKit files	
2. Run KioskKit shortcut using Powershell	3. Elevate administrator prompt
4. Accept all (A) elevation prompt	5. Execute sub-scripts
6. Restart Computer	
Extensions (aka Alternative Flows)	
At line 5, if error then inform user and exit. User inform supervisor and troubleshoot error according to log file.	

Figure 5

Only InfoSec or ACAD interns have direct interaction with the KioskKit. In other words, Kiosk end users have no interaction with the KioskKit or PC setup as they receive a PC already setup and plugged in for them at their workstation.

InfoSec and ACAD interns are expected to have a basic understanding of how to use a computer. However, they are not expected to have any programming knowledge. Therefore, the KioskKit must require as little user input and understanding as possible. Additionally, the script must inform the user of the script's progress in an easy to understand way. The KioskKit was designed to only require that the user run the script from the filesystem. Referring to Figure 6 below, the Intern applying the KioskKit would initially install the OS, login, and locate and then run the KioskKit script.

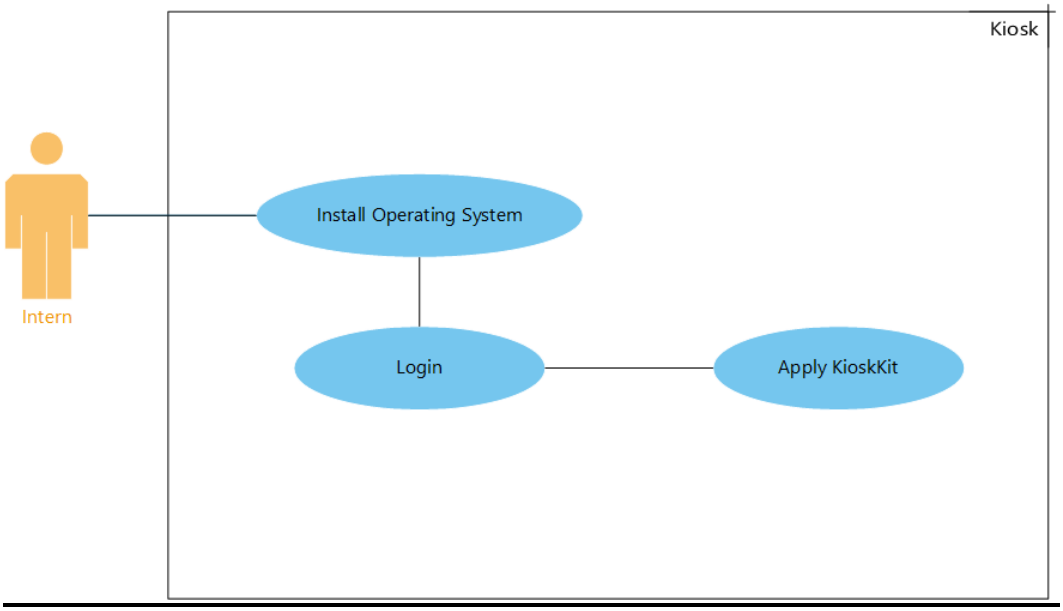

Figure 6

Costs and Benefits

Currently, it would take an intern an average of 30 minutes of time to perform manual hardening. This time would vary depending on how long it takes to run Windows Updates. Interns are expected to have follow guidelines set by the existing hardening policy and proceed through a long list of steps. However, interns are not expected to know how to program. An intern makes $10 an hour so hardening costs an average of $5 a machine which can be somewhat reduced by working on a few machines between longer steps.

The proposed system will remove the need to perform the steps manually while saving some time that would be spent navigating the system. It takes KioskKit an average of 10 minutes to perform hardening. Therefore, it saves around 20 minutes per machine and can be run on any number of machines at once with minimal input from the intern. This means that KioskKit saves about $4 each time it is used.

Task Name	Time Spent (min)	Cost ($25/hr)
Project Proposal	45	$ 18.75
VM Setup	20	$ 8.33
GroupPolicy Script	130	$ 54.17
Project Plan	45	$ 18.75
Interim Presentation	90	$ 37.50
Design Documents	30	$ 12.50
Registry Script	230	$ 95.83
Interim Report	160	$ 66.67
Local Security Policy Script	60	$ 25.00
Task Scheduler Script	160	$ 66.67
Hardening Standard Flow Chart	20	$ 8.33
User Settings Script	100	$ 41.67
Installs and Updates Script	140	$ 58.33
Core Script	160	$ 66.67
Presentation	120	$ 50.00
*No recurring costs	**Total**	$ **629.17**
	Scripts only Total	$ **408.33**

Figure 7

Referring to Figure 7 above, the cost of development comes to around $408.33 for the scripts alone and $629.17 for the total cost of development including planning and documentation. This is assuming that the hourly development cost is $25 an hour. There are no recurring costs for using KioskKit that are directly linked to KioskKit's use. External tools such as Qualys, Cisco AMP, Windows 10 LTSC, and CIS-CAT are used by the University of South Alabama and is easily obtainable by interns using KioskKit. Chrome and LGPO are downloadable free from their respective sources.

Lessons Learned

From this project, I learned a few things about the Windows operating system. First, I learned that not every Local Security Policy has a Registry key. This made it tricky to use the Registry to modify Local Security Policy settings, and it limited what I could do with it because there is no other way to automate Local Security Policy changes except through the Registry key. Next, I learned that Windows does not like user accounts not having a password. It will proceed to prompt the user to provide a password even when the user account is configured to not require a password. Later, I used a work-around to re-apply the setting after the user account was created and it seemed to fix the problem. PowerShell turned out to be very effective at automating Windows tasks allowing for all the steps in the hardening process to be automated. Lastly, I learned how to prevent a specific user from executing programs. This was useful for the UserSettings sub-script to prevent the user from accessing administrative tools that could be used by attackers to gain access to the system. Before, I had only thought it was possible to block all non-administrative users; however, the script allows for it to apply only to specific users.

Broader Impact

The classes that helped me with this project the most were Human Computer Interface, Needs Assessment, and Project Management. Human Computer Interface (HCI) helped me create effective Use Case Diagrams. The class helped stress the importance of including only necessary steps and keeping it simple. Also, each task should lead with an action to make it clear what the actor is doing. Needs Assessment helped with the evaluation of user needs as well as reinforcing what I learned in HCI. It helped me create effective Use Case Narratives to describe what is happening for a specific task. Project Management helped me create effective project timelines and managing time wisely throughout the project. Specifically, I learned the danger of putting off the bulk of the work towards the end of the project. Therefore, my project plan was designed around working on one script a week. Some scripts ended up taking longer due to complications in development or when a project activity was due; however, I finished the project on time with a complete product.

Conclusion

KioskKit was a successful project that saves interns time when performing hardening. Additionally, the project ended up being used by the Information Security Office. This project gave me the opportunity to practice my project management, needs assessment, and programming skills. Furthermore, it gave me the opportunity to explore how the Windows 10 OS works and provided me with additional knowledge that can be applied to my job. In conclusion, my bosses and I are satisfied with the outcome of this project.

References

CIS-CAT Pro. (2019). Retrieved December 9, 2019, from https://www.cisecurity.org/cybersecurity-tools/cis-cat-pro/.

Cisco Advanced Malware Protection. (n.d.). Retrieved December 9, 2019, from https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/index.html.

Google Chrome. (n.d.). Retrieved December 9, 2019, from https://www.google.com.

Information Security. (n.d.). Retrieved October 20, 2019, from https://www.southalabama.edu/departments/csc/informationsecurity/.

Margosis, A. (2016, January 21). LGPO. Retrieved from https://blogs.technet.microsoft.com/secguide/2016/01/21/lgpo-exe-local-group-policy-object-utility-v1-0/.

PCI DSS. (n.d.). Retrieved December 9, 2019, from https://www.pcisecuritystandards.org/.

Qualys. (n.d.). Retrieved December 9, 2019, from https://www.qualys.com/.

Wilcox, J. (2018, November 29). LTSC: What is it, and when should it be used? Retrieved October 20, 2019, from https://techcommunity.microsoft.com/t5/Windows-IT-Pro-Blog/LTSC-What-is-it-and-when-should-it-be-used/ba-p/293181.

Windows 10 LTSC. (n.d). Retrieved December 9, 2019, from https://docs.microsoft.com/en-us/windows/whats-new/ltsc/.

Virtual Box. (n.d). Retrieved December 9, 2019, from https://www.virtualbox.org/.