The Dark Blue CyberPatriot Training Tool

Scott Semian, Keith Harrison
The University of Texas at San Antonio
scott.semian@my.utsa.edu, Keith.Harrison@utsa.edu

Abstract

The Dark Blue CyberPatriot Training Tool was developed to help better prepare teams for the CyberPatriot competition, which aims to introduce middle school and high school students to the field of cybersecurity. Dark Blue allows coaches to create a configuration file for a practice image, with a generator component that will take the file and attempt to create as much of the configuration as possible automatically on a blank virtual machine image. To simulate the competition round, Dark Blue also includes a custom built scoring engine that is configured by the generator. We believe Dark Blue improves ease of use significantly over other training tools, and could help teams create better practice images with less effort and knowledge required. It also allows teams to create and share practice images freely with other teams, which we believe could have a lasting positive impact on the competition as a whole. Dark Blue is to be released as free and open-source software in the near future.

1. Introduction

Cybersecurity is quickly becoming one of the most critical areas of focus for many companies and governments around the world as various industries continue to adopt more digital solutions and more facets of life continue to move online. Cyber attacks against large corporations, critical infrastructure, and government entities are becoming increasingly common [1]. Many of which could have had potentially devastating consequences for hundreds of millions of people, like the 2017 Equifax hack which resulted in the names, birth dates, and Social Security numbers of an estimated 147.9 million US citizens being stolen, leaving them potentially vulnerable to identity theft [2]. As internet connected services, internet of things devices, and other digital communication devices continue to grow in popularity, availability, and capability, the frequency and severity of attacks will continue to increase. Organizations and governments have started to fully recognize the threat they are facing, and are therefore beginning to take the necessary steps to properly address the problem.

In this paper, we will discuss the need for additional talent in cybersecurity roles in industry as well as government, discuss the CyberPatriot National Youth Cybersecurity Competition, and introduce the Dark Blue CyberPatriot Training Tool along with its features and how this will help teams better prepare for the CyberPatriot competition as well as our thoughts on how the availability of an easy-to-use, free and open-source training tool may impact the CyberPatriot competition.

2. The Need for Cybersecurity Workers

In light of escalating attacks, every organization that relies on computer systems will need some amount of cybersecurity expertise, whether they are employees or contractors. Large organizations with sensitive data, like big data companies and government agencies, must adopt rigid security postures and form entire departments to monitor and secure assets. This means companies will be looking for employees with experience in the field of cybersecurity, something that many sources indicate will become increasingly difficult as the disparity between the number of jobs and individuals to fill them continues to grow. One such source estimated in 2019 that there would be 3.5 million cybersecurity positions vacant globally [3]. Another report from CBS News based on Cyber Seek data indicated there were slightly less than 500,000 vacant cybersecurity positions in the United States as of May 2021 [4].

In the last few months at the time of writing, we've seen attacks against government agencies and companies managing critical infrastructure that have had significant impacts on US citizens. In May of 2021, the operator of the Colonial Pipeline faced a cyber attack which lead to a temporary shutdown of the system, causing panic buying and eventually shortages

URI: https://hdl.handle.net/10125/79618
978-0-9981331-5-7

of gasoline along the Eastern Seaboard [5]. Around the same time, the city of Tulsa, Oklahoma was facing a similar ransomware attack which left many city services inoperative, including calls, dispatching, and mapping systems for the local fire department [6]. In response to the Colonial Pipeline attack new regulations are being introduced that would require pipeline operators to report exploits to some agency of the Department of Homeland Security (DHS), as well as providing a "cyber official" from the company that is available 24/7 to communicate with federal officials. Previously, this guidance was issued voluntarily by the DHS, but this new directive will be mandatory [7]. President Biden also issued an executive order establishing the Cybersecurity Safety Review Board and pushing for higher standards of security for government bodies [8].

Increasing cybersecurity legislation and the attempt to set baseline security standards for government agencies and critical infrastructure providers will likely mean an increased demand for cybersecurity workers in these industries. While these efforts in the US are currently focused on the federal government as well as companies managing some sectors of critical infrastructure, it is possible in the future we could see these expand to other sectors, and potentially to companies that store and process large amounts of data on US citizens, like Alphabet, Amazon, or Facebook.

Some government agencies handle information that is classified, which requires strict security policies and constant monitoring to ensure it is not compromised and disclosed, meaning they must employ security experts. The Covid-19 pandemic has underscored the possibility that government business may not be able to be conducted from physically secured offices, potentially requiring employees to complete work and handle critical information from unknown and possibly unsecured environments, meaning that security professionals working for these government agencies must be more adaptable. Additionally, government agencies would be extremely high value targets for nation-state level attackers, with each successful attack carrying potentially detrimental ramifications for the agency or the government as a whole.

3. The CyberPatriot National Youth Cyber Defense Competition

In response to the deficit in workers in the cybersecurity field, the CyberPatriot National High School Cyber Defense Competition was created, with the first competition round in 2009 serving as a proof of concept for the idea [9]. Since then, the competition has been incredibly successful, going from 300 schools expressing interest in the second event and 170 participating to over 6,700 teams registering for CyberPatriot XII in 2019 [10]. The program has also been expanded to include middle schools as well [9]. The overall goal of the CyberPatriot program is to help address the shortage in the workforce, specifically in the United States, by introducing students to the field of cybersecurity before they've entered college to encourage more students to enroll in programs that focus on security and to cultivate more interest in the field.

Competition rounds consist of a set of preconfigured virtual machine images for Windows and Linux which contain vulnerabilities or misconfigurations the competitors must address without any information given as to exactly what those issues may be. Therefore, the competitors must read the scenario for that specific round and extrapolate which security policies and best practices must be applied to that image to gain points. The scoring engine will continuously monitor the the image and award or deduct points based on certain actions that are taken on the virtual machine by the competitors, providing real-time feedback on their performance as they compete. Teams generally consist of 5 members, with coaches and mentors to help guide and instruct the teams outside of official competition images, when the team is supposed to compete independently.

This concept has proven successful and has been adapted internationally for competitions in other countries, like CyberTitan in Canada, CyberCenturion in the UK, CyberTaipan in Australia, and more. All of these competitions are modeled after the CyberPatriot competition, and share the same goal of exposing school aged children to the field of cybersecurity in order to increase the size of the talent pool in their respective countries.

Many organizations and governments around the world use the CTF, or Capture the Flag system, in order to test organizational security practices. CTF exercises can be used for training and education depending on the overall goals and specifications of the contest, but prior to 2009, there are not many instances of capture the flag activities that were geared towards school aged children. CyberPatriot, while somewhat similar to a CTF in concept, aims primarily to educate middle and high schooled aged children and introduce them to the field of cyber security and encourage them to enter cyber security roles, where later in their careers they would encounter more traditional CTF scenarios.

4. The Need for a Training Tool

In order to train and prepare for official competition rounds, most teams will be heavily reliant on their coaches and mentors, who might have expertise in the cybersecurity field or previous experience with the competition. Outside of that, there are a number of slide decks provided by the Air Force Association which cover a number of operating system and cyber security basics. The only ones currently available publicly are archives of previous seasons' modules [11].

Texas A&M University-Corpus Christi released a tool which allowed for coaches to configure a scoring engine that was developed to mimic the official competition image, and supports Windows 7 and 8.1. The tool was developed on a grant from the National Science Foundation for the CyberPatriot competition [12].

The CyberPatriot competition also provides a certain number of practice images every season to prepare the teams for upcoming rounds of competition. The practice rounds behave the same way as the normal competition rounds, and like the official rounds answer keys are never provided. A few years ago, an additional round was introduced prior to the practice round containing some of the most basic vulnerabilities that appear in competition images. This round includes an answer key and serves primarily as an introduction for new teams.

Apart from these options, teams must create their own methods or materials to train and prepare for the competition.

From personal experience with the CyberPatriot competition during high school, it can be hard for a team to improve if they don't have consistent access to coaches and mentors. For the majority of my own time during the CyberPatriot competition, we did not have any coaches with experience in the field or with the competition, so during each competition round we only improved by luck or from experience gained in previous rounds.

When there were no practice images available it was difficult to practice in order to improve and score higher in the next round of competition, especially since we never knew what we were missing.

The tool from Texas A&M University-Corpus Christi (TAMU-CC) provides some functionality to replicate a practice or competition round, and is available on the CyberPatriot website, but there are a few notable disadvantages with this tool which Dark Blue seeks to eliminate. The versions of Windows that the TAMU-CC tool supported are limited, and it doesn't include Linux support at all. Any vulnerabilities that were supposed to be resolved would have to be manually created before the scoring engine is configured, which raised the technical requirements to use the tool above which some teams might be comfortable.

5. Functionality

The goal in developing a new training tool for the CyberPatriot competition was to create a method that would allow a coach with a relatively small amount of technical knowledge to create a basic practice image for their team to complete, which would function as similarly to an official image as possible. To achieve this, the complexity of using the tool should be low, with as many operations automatically performed as possible.

Therefore, the user should be able to configure their practice image, load some files onto a virtual machine, and then the training tool would attempt to generate and configure as much of the image as possible to match that specification. Unfortunately, not every component can be automated, but the goal was to make a tool that could perform enough of the image configuration so that the tool is still usable regardless of skill level. The desire was that the most complex aspect in using Dark Blue would be creating a blank virtual machine itself, something that can hopefully be overcome with video and written tutorials.

In order to help a team better prepare for a competition round, the tool should match the feel and functionality of the official competition scoring engine as closely as possible. Because the official scoring engine is proprietary, and could not be used for an application like this due to a desire to keep the engine secure, the scoring engine would have to be written from scratch.

Another key requirement for Dark Blue is that it support both Windows and Linux, with feature sets that are as closely matched as possible, though they may not be identical.

In order to ensure the longevity and utility of the training tool, the plan from the beginning was that it would eventually be released completely open-source so coaches and competitors can continue to iterate on and improve the tool for years to come.

Dark Blue consists of three main components that are each designed to help individuals create simulated CyberPatriot practice images. The first component, or tool, is the front end system which allows the individual creating the image to specify what should be scored, which will be discussed in more detail later, but for example would allow them to specify the names of the users that should be present on the system, as well as whether or not they should be allowed on the system and have administrative privileges. This produces a JSON

file specifying the configuration of the image, which will be loaded onto a blank virtual machine image that the user creates, where the next tool, the generator, will take that file and attempt to automatically create users and give them permissions based on the file. The generator will also configure the scoring engine, which will then be run to allow competitors to simulate a round of the CyberPatriot competition.

5.1. Languages and Technologies Used

The first component to consider in the development of Dark Blue was what languages and technologies should be used. The tool was roughly divided into three parts: the configuration tool to create an image, the generator which would match a blank virtual machine to that configuration, and the scoring engine. Various languages were evaluated based on their flexibility, capability to perform all the necessary tasks for the component, and their ease of use and readability. The idea being that a high school level competitor who might have some technical knowledge from experience with CyberPatriot, but without previous programming experience would be able to read and understand the code in order to make a small change. In order to keep complexity down, a language would need to be chosen that works natively on both Windows and Linux and is capable of performing all of the tasks required.

The choice was made to use Python for both the scoring engine and the image generator. Python was chosen because it is a popular modern language renowned for its flexibility as well as being one of the more beginner-friendly languages. This would require various system level utilities, as well as a single user interface for the user to load their configuration file, start the generator, and start the scoring engine which is currently being redesigned to provide a better user experience.

Electron was chosen as the platform for the image configuration tool, using ReactJS as the framework to create the app. Electron is a popular platform for desktop web apps based on the Chromium rendering engine, and ReactJS is one of the more popular JavaScript frameworks currently. This approach was chosen rather than using UI implemented in Python because this user interface was much more complex and dynamic, as opposed to the back end interface which consisted mostly of a few buttons. In order to reduce complexity and minimize the number of actions that must be performed on a virtual machine, the decision was made to use some kind of intermediary file between the image configuration tool and the generator. This could facilitate easier sharing of simple practice images, and doesn't require the user to spend as much time interacting with a virtual machine than if they configured the image directly on the virtual machine with a single tool that would then apply it.

The JSON format was chosen for this intermediary file, as it is relatively ubiquitous and easily supported by both JavaScript and Python, as well as being relatively human readable and editable if an individual preferred that approach to creating and editing their image. In order to use the tool, a coach or team member could simply download the zip file, unpack it, then use the configuration tool to produce a JSON file containing the specifications for the image, which would be loaded onto a virtual machine along with a compiled generator/scoring engine application. Having these components all compiled to executable formats was desired to eliminate the need for installation of prerequisite packages or technologies.

5.2. Scoring Engine

The scoring engine was the first component to be implemented, and is written entirely in Python with some calls to shell commands. Since the scoring engine has to be cross-platform, a single main scoring engine class is used with operating specific helper functions. There are a few common functions that are implemented identically on both platforms, which are located with the scoring engine. These are mostly related to file paths, scoring question files, and generating the scoring report. Helper functions are only loaded for the appropriate operating system, so the scoring engine can leverage operating system specific packages and utilities.

The primary responsibility of the scoring engine is to run indefinitely in the background, scoring all aspects of the image on a fixed timer interval. There are some other functions that it most also perform, like updating the scoring report, which is a simple HTML file that is created on the desktop of the user, which is populated with messages to inform the user why they have gained or lost points as well as their total score, similarly to the official competition system. It also handles issuing desktop notifications when the user gains or loses points, which are tracked using a unique scoring ID field that is assigned to each object when they are created by the generator.

This approach was necessary to simplify the process of repeatedly scoring the same items without issuing additional notifications for items that had already received one, while accounting for the possibility that an item that has already received a notification for gaining points might later require a notification for losing points, which could repeat indefinitely. The scoring IDs are

kept in either a positive or negative list, and after a notification has been issued they can only move between the lists to generate a notification, and cannot generate two positive notifications for example.

After the image is scored, the scoring report is updated to include all scoring messages that have been generated. Then the scoring engine is saved to disk using Python's pickle module. Saving is necessary because in the event the virtual machine is restarted or crashes, the image should be able to be resumed flawlessly. If the image was not saved, the generator would have to be run again to create the scoring engine object, as well as populating various other objects related to the configuration, which might not complete successfully because certain modifications have already been made to the operating system by the previous run of the generator. Allowing the generator to be run twice while still creating objects for the scoring engine was deemed too complex, so instead the scoring engine is continuously saved so it can be easily resumed. When the back end program, which contains both the scoring engine and generator, is run it will first automatically check to see if an image of the scoring engine is available, which will then be resumed.

The back end generator and scoring engine components share a single user interface, which at the time of writing has been removed and replaced by a command line interaction with the user while the user interface is redesigned. After the redesign is complete, the user will have the option of using the command line version or the graphical version to start the generator and scoring engine.

5.2.1. Windows Much of the implementation for the Windows scoring engine leverages the pywin32 package, which provides a method of directly using the Windows API within Python.

To score users, a list of users is created by querying for the users in the 'Users' and 'Administrators' groups, along with their user SID to ensure that a user is not accidentally deleted and then recreated with the same name to recover points. Decisions to award or remove points are then made based on the user object that was created by the generator and its fields.

Users can be scored for whether they're authorized to be present on the system and if they're supposed to be an administrator. Users are not usually created during a competition, so points are only awarded if a user who is not authorized to access the system is deleted. A user can either require that they are removed from the Administrators group, or they can be added in order to gain points. The competition 'README' file will outline which users are allowed on the system and which users belong in the Administrators group.

All three Windows firewall profiles can be scored for either enable or disable, though generally the only transition that would gain points in a competition image is for a firewall profile to be enabled. This is scored using a subprocess call to the shell 'netsh advfirewall' command, as a more reliable method of determining firewall state using pywin32 was not available. The returned output of the command is split and parsed to get the 'OFF' or 'ON' values that are printed. The output of the command is relatively stable, so this solution was deemed acceptable until a better method is found.

To score services, the Python psutil module is used to search for a service, but is limited to searching by the actual service name rather than the common or display name. This is something that is in the documentation, as well as on the page for configuring services in the configuration tool. The example given is that Remote Desktop Services would be scored by searching for 'TermService'. Services are scored by their running state, with scoring of the startup types currently being implemented.

The most important feature in the Windows portion of the scoring engine is the registry scoring feature. Currently, arbitrary registry entries can be scored from either HKEY_LOCAL_MACHINE or HKEY_CURRENT_USER can be scored with their full path, starting at the HKEY root. This is more of an advanced feature that should allow coaches with advanced technical knowledge to create more complex images which might approach difficulty levels of later rounds of competition. Currently the scoring engine is capable of automatically scoring registry keys with string, hexadecimal, and decimal values.

The final component of the Windows scoring engine is scoring installed programs. This is done by reading registry keys in the Uninstall key, which somewhat limits the number of programs that can be scored as it only supports programs that create and remove that registry key.

5.2.2. Linux Linux contains many of the same features as the Windows scoring engine but is implemented slightly differently, using more shell commands than the Windows implementation.

User scoring makes use of the pwd module to get a list of users and their user ID, which is mapped into a dictionary. Sudoers are listed using a shell command with their usernames saved into a list. The scoring transitions that are possible are identical to those on Windows.

Processes are scored by creating a list of all currently running process names using psutil, which has an iterator function to enumerate all processes on the system. Points are then assigned based on whether or not a name is found in this list and if it should appear based on the configuration file.

Packages are scored using dpkg to list all packages on the system and using grep to search for a specific package name. Points can be awarded for packages being added or removed, and subtracted for the same reason given the contents of the configuration file.

Similarly to the Windows registry, the most advanced and arguably the most important feature is the 'configuration file' scoring. This scores arbitrary entries in the proc filesystem. This allows for configuration and scoring of many different operating system level features, for example, disabling address space layout randomization and scoring to see if it is re-enabled at any point. Like the registry, this feature is going to be utilized most effectively by individuals with more technical knowledge.

5.3. Generator

The generator is responsible for taking a configuration file and matching the system to that file as closely as possible. To run the generator, a user must select a file path to a JSON file using the back end user interface and then clicking the button to start the generator. The generator usually takes under a second to run, with an exception with some of the commands taking a few seconds before they apply. Disabling firewalls in Windows is the most notable example.

Arguably the most important component of the Dark Blue CyberPatriot Training Tool overall is the generator. With this inclusion, a simple practice image can be created without much technical knowledge and configured automatically. Most of the simple features of the scoring engine are those that are capable of being automatically configured by the generator, like users and file paths. If Dark Blue only contained a scoring engine and a method for configuring it, it might require more technical knowledge than some coaches might have because they would have to create all of the users and file paths and ensure they are properly configured.

While Dark Blue still has some limitations, it is our belief that the generator has a significant positive affect on the ease of use and accessibility of the training tool.

5.3.1. Windows The Windows generator can automatically configure users, including creating new users with a given name and setting their administrative privileges appropriately, as well as then reading the created user's SID and storing it with the user object that is created for the scoring engine. The SID is stored to make sure that deleted users cannot be re-added to regain points that have been lost. All of these functions are performed using various components from the pywin32 module.

Firewall profiles can be automatically enabled or disabled, which is done using the Set-NetFirewallProfile Powershell command. While functional, this approach is not optimal as it takes between five and thirty seconds for the firewall to actually change states after the command executes. The generator will complete and return before this happens, so if the scoring engine is run before the firewall has actually changed states points can be awarded falsely, but will then be removed the next time the scoring engine runs after the state change has occurred. This is considered to be a minor bug, as the issue will automatically correct itself, and the documentation recommends to wait a minute between running the generator and taking a snapshot of the virtual machine to distribute or running the scoring engine. This is preferred to altering the generator to wait until the firewall has changed states or altering the scoring engine to skip scoring firewalls until after a certain number of runs of the scoring engine.

To configure the registry, the generator will first check to see if a key path exists, and if not it will create the key and set it to a default value that is specified in the configuration file. If the key already exists and the create flag is set for that entry, the generator will attempt to modify the current value of the key to match the specified default value. The modification of existing registry keys is still being tested, as it is difficult to make a system that can reliably modify arbitrary registry keys, especially when the data type of the key is unspecified. Adding functionality for the user to indicate the type of key has been considered, but will not be done until a new version of the registry key page is created that would better allow for more fields. It has also been observed that some services might automatically reset their registry keys if they are modified, so it's possible the desired value of a key might not be maintained unless certain services are disabled, which is unlikely to be fixed in future versions of Dark Blue as it might not be possible or might render the virtual machine unstable.

The generator does not modify installed programs or services. Due to the differences in installation processes for each program, as well as the complexity of automatically locating and retrieving the correct installer from the internet, it is unlikely this process will be automated in the future. Installing programs is likely something a majority of users of Dark Blue would be

able to do, so as long as it is adequately communicated to the user that they must install programs prior to running the scoring engine, this solution has been deemed acceptable.

Automatically modifying services is something that will likely be added in the future. To configure services, the generator would simply have to start or stop them, as well as changing their startup type, which should be possible using some Python module or the Windows API.

Both Windows and Linux have identical functionality for creating file paths, question files, and the 'README' file on the user's desktop. It will first search for the file path to see if it exists, and if not it will then attempt to create the path along with any subdirectories in the path that might be missing.

5.3.2. Linux Linux is very similar in terms of the capability of the generator, but has some advantages considering the fact Linux is much more command line based than Windows.

User configuration is the same, with users being created and added to the sudoers group if necessary. This is done using a shell call with the useradd command, with users then being added to sudo with usermod. Their user ID is also recorded for the same reason that their Windows SID is saved – to ensure that a deleted user can't be re-added.

Process configuration has a notable advantage over Windows services in its ability to create a dummy process with the given process name automatically. If the create dummy option is selected, the generator uses a simple Perl command that holds indefinitely in the background. While the process doesn't do anything, it will still appear in the system process list with the name specified in the configuration file.

Packages have the same limitation as Windows programs, requiring manual installation. It's possible this will be automated, if the name of the package that appears in dpkg matches the name that would be used by aptitude to install it, or another field could be added to the configuration tool which allows the user to specify the name used to install the package. This would likely be more useful than automated Windows program installation as installing packages on Linux might be more difficult for inexperienced users.

Similarly to the Windows registry, the Linux generator can modify files present in the proc filesystem to change their default values to match a specified value. However, arbitrary files cannot be created unless they are located outside of /proc. It is theoretically possible that a system could be developed by which kernel modules are created to create arbitrary interfaces in /proc, but this solution will not be investigated further unless it is highly requested by the community. The generator can create configuration files at an arbitrary file path, as long as they are not inside /proc.

The Linux generator uses the same functions for creating file paths and question files as the Windows generator, which will create and necessary subdirectories in the path.

5.4. Configuration Tool

The configuration tool allows the user to create a new virtual machine image configuration. This configuration is stored as a JSON file, which will be used by the generator to configure the virtual machine, as well as to create and initialize the scoring engine.

The intention is that the configuration tool will be used to create a new image outside of the virtual machine, and then just the JSON file as well as the compiled scoring engine and generator program loaded onto the virtual machine in order to minimize the amount of time that the user has to interact with a virtual environment. This is preferred because of the relatively high hardware requirements for running a smooth virtual image.

When the user launches the configuration tool, which when built has both portable and installer variants, they are first asked to either select an existing image or create a new image configuration. Either option opens the system file browser, either to select an existing file or to specify the path to create the new file.

Then, the user moves to the homepage which contains some basic user information, like the path to the file they're currently editing, as well as the operating system. The home page contains fields for overall image configurations, like the number of possible points, as well as the readme file, which can be automatically generated.

Navigation through the configuration tool is done using the sidebar, which contains tabs for each category for the given operating system. The sidebar is different for Linux and Windows, with options only being displayed for one operating system. Each page is automatically loaded from the file when the user navigates to it, and is automatically saved when the user navigates away.

Pages are formatted as tables with text fields and checkboxes being the primary method of configuration for each item. Almost everything for each operating system is a list of individual objects, with objects being added or removed by the user which creates a blank entry of that type.

After the image is configured, the user can use the answer key generator which will attempt to produce an answer key for the image based on the current configuration settings.

Currently, the configuration tool has only been built and tested for Windows, but given the cross-platform nature of Electron, it is possible a Linux build for the configuration tool can be introduced in the future.

6. Initial User Feedback

Dark Blue was demonstrated to a high school CyberPatriot team in late January 2021, consisting of both coaches and competitors, with positive reception. The group stated that the ease of use of Dark Blue far exceeded that of their current training toolset which was much more complicated to set up and configure. They were provided with an alpha build of the tool for testing and to generate feedback. The response was positive, there were some bugs that required fixes, with a focus on expanding the documentation and tutorials that were available, which at the time had been written specifically for that initial test version.

There were also several requested features, some of which have already been added and some of which are in consideration or being evaluated for feasibility of implementation into an automated tool, one of these such features is adding the ability to create hidden users. This is currently being evaluated to determine if it is possible to do using the current Python packages that are used to create users.

In terms of overall functionality, the team was pleased with the ease of use and the low difficulty in creating simple practice images. While the tool has yet to be officially released, we look forward to a public release with more teams having the ability to try it out and provide feedback to improve the feature-set and user experience.

7. Comparison to Other Tools

The main tool that Dark Blue will be compared against is the Scoring Engine Tool v3.0.1 from Texas AM University-Corpus Christi which is available on the CyberPatriot website, which will be referred to as 'SET'.

The Scoring Engine Tool supports scoring of more individual features than Dark Blue currently does, but it does not perform any configuration of the image. SET is designed to be used in a fundamentally different way to Dark Blue, where the coach will create a virtual machine, create vulnerabilities and fully configure the VM environment, and then load SET onto the machine. SET will then pull lists of various components, like users and installed programs, and allows the user to select which ones to score. This has some advantages, as it allows users to more easily select certain elements they want to score, like installed programs compared to Dark Blue which asks them to find the exact display name of the installed program, but it still requires users to have the program installed prior to running SET.

There are other features that could be implemented in Dark Blue in the future, like group scoring, firewall rules, start up programs, and scheduled tasks that SET supports, but all of them require manual configuration from the user to be performed prior to configuring the scoring options in the Coach Configuration Tool.

One of the most notable difficulties in using the Scoring Engine Tool is it's limited support for operating systems. The tool only supports Windows 7, 8.1, and Server 2008. SET does not run on Windows 10, and it is extremely difficult to create a Windows 7 or 8.1 image compared to creating a Windows 10 virtual machine. Both Windows 7 and 8.1 *require* a license key in order to either obtain an ISO from Microsoft, or to install the operating system in a virtual machine. If teams don't happen to have a license key that is valid, they may not be able to create the virtual machine at all, or might be relying on downloading ISOs from unknown parties on the internet, which is a security risk.

Windows 10, on the other hand, allows users to create an ISO for free with the Media Creation Tool from Microsoft, and Windows 10 can be installed and used indefinitely without a license key, although some features might be disabled. With the fact that Dark Blue also supports Linux, we believe this is a major advantage of Dark Blue.

Another limitation of SET is that, to the best of our knowledge, the source code was never released and the tool is no longer supported or maintained, so these issues will likely never be addressed. With Dark Blue being a tool that will be released completely open source, it can be maintained by the community indefinitely to add new features, fix bugs, and adapt to changes in operating systems and the competition itself.

8. Impact

While Dark Blue has not yet been publicly released at the time of writing, we believe the Dark Blue CyberPatriot Training Tool will prove to be a valuable tool for teams of all levels to improve in the CyberPatriot competition when official practice images are not available. While scoring information and answer keys are not released, it is likely the bottom few percent of teams that participate in the competition do not score

all of the points available in that competition round that Dark Blue can replicate. With additional practice using Dark Blue, as well as images configured by other teams, teams lower in the standings might be able to see significant improvement in their scores as they learn more about the operating system, hardening, and the competition.

If Dark Blue as a training tool is powerful and flexible enough that higher level teams are also interested in using it to create practice images, our hope is that they would be willing to create and share practice images for any team to use which would create a community around Dark Blue which could eventually lead to an overall increase in the level of competition. While not an intended use case, it's also possible that Dark Blue could be used for computer security awareness and training programs targeted at individuals without a technical background of security experience.

In private demonstrations with teams so far, coaches and participants have expressed excitement at the feature set and ease of use of Dark Blue, stating that it is far easier and faster to use than their own custom solutions for creating practice images. With an alpha build of Dark Blue, teams have been successful in creating custom images with which to practice for competition rounds.

We look forward to the public release of Dark Blue and seeing how teams utilize the tool and expand their knowledge of computer security.

9. Future Work

As previously stated, Dark Blue is still a work in progress tool that is closed source with only a single private release alpha, as of June 2021. The first public release version is targeted for summer 2021, with the open source release to follow. For the first public version, there are a number of features and improvements to be implemented, as well as further testing to improve stability and functionality. One aspect is to improve error handling and malformed inputs from the configuration file for both the generator and the scoring engine to make the experience as seamless as possible.

One of the largest components that will be changed is the implementation of a new user interface for the entire back end system using Kivy, which will greatly improve the user experience and make the process much clearer. A refactor of much of the code base, particularly the scoring engine and generator systems, will add comments and make the code much clearer and easier to understand ahead of the open source release. Another component to this is writing tutorials and contribution guidelines to keep the code base clean and consistent as other developers may start contributing, as well as to provide help for any competitors with no or limited programming knowledge who might be interested in exploring the source code or making small changes.

Additional documentation is needed, including complete guide from start to finish on how to download and use Dark Blue, create a virtual machine image, load files onto the virtual machine, and running the generator and scoring engine. This will make it more accessible to more competitors and coaches without significant technical experience. Part of this includes creating video walkthroughs that will demonstrate the process, though these would be updated much less frequently than written guides.

After the open source release of Dark Blue, we don't know what the future will hold for the project. We hope widespread adoption and use of the tool can create a meaningful difference in the CyberPatriot competition and help the program achieve its goal of educating the next generation of cyber security experts in the United States. With the project being open source, students, coaches, and other developers can work to improve usability, add new features, improve automatic image generation, and more. The desire is that Dark Blue will primarily be community maintained, with less frequent commits from the original developers as time goes on. We believe this, combined with ease of use improvements over other training methods, as well as both Windows and Linux support, could make Dark Blue an integral tool for many teams' training regiment for years to come.

10. Conclusion

The CyberPatriot competition is an important component in strengthening the United States' cybersecurity workforce, and we believe Dark Blue will prove to be a useful tool for teams seeking to improve and practice for the competition. Dark Blue allows coaches and competitors to configure a practice image which will be used to automatically configure a blank virtual machine to match the configuration as closely as possible, with a fully custom scoring engine that will allow teams to simulate a competition round. The first public release is planned for the summer of 2021, with an open source release to follow. We believe Dark Blue has the potential to make a significant positive impact on the competition and its competitors and will assist the CyberPatriot program in achieving its goal of training America's next generation of computer security professionals.

References

[1] "Significant Cyber Incidents | Center for Strategic and International Studies." https://www.csis.org/programs/strategic-technologies-program/significant-cyber-incidents, May 2021.

[2] B. Barrett, "How 4 Chinese Hackers Allegedly Took Down Equifax." https://www.wired.com/story/equifax-hack-china/, Oct. 2020.

[3] S. Morgan, "Cybersecurity Talent Crunch To Create 3.5 Million Unfilled Jobs Globally By 2021." https://cybersecurityventures.com/jobs/, Feb. 2018.

[4] K. J. Brooks, "U.S. has almost 500,000 job openings in cybersecurity." https://www.cbsnews.com/news/cybersecurity-job-openings-united-states/, May 2021.

[5] A. Ma, "Gas outages spread further across the US even after the hacked Colonial Pipeline began pumping again." https://www.businessinsider.com/gas-shortages-spread-across-us-colonial-pipeline-restarts-2021-5, May 2021.

[6] J. Cooper, "Multiple City Of Tulsa Services Disrupted After Cyberattack." https://www.newson6.com/story/609af1abd9fc770bae4e1ffe/multiple-city-of-tulsa-services-disrupted-after-cyberattack, May 2021.

[7] K. Canales, "DHS will reportedly introduce the US's first cybersecurity regulations after the Colonial Pipeline hack revealed the fragility of American infrastructure." https://www.businessinsider.com/colonial-pipeline-hack-dhs-cybersecurity-regulations-tsa-washington-post-2021-5, May 2021.

[8] S. Mucha, "Biden issues executive order following mounting cyberattacks." https://news.yahoo.com/biden-issues-executive-order-following-223045960.html, May 2021.

[9] G. B. White, D. Williams, and K. Harrison, "Developing a national high school cyber defense competition.," *Proceedings of the 14th Colloquium for Information Systems Security Education*, pp. 83–89, 2010.

[10] "National youth cyber defense competition registration report 2019-2020." https://www.uscyberpatriot.org/Documents/Fact%20Sheets/2019-20%20CyberPatriot%20XII%20Registration%20Report.pdf, 2020.

[11] "Archived Training Modules." https://www.uscyberpatriot.org/competition/training-materials/training-modules.

[12] "Build Your Own Practice Images." https://www.uscyberpatriot.org/competition/training-materials/practice-images.