How To Win CyberPatriot

Posted **Aug 8, 2023**

By **Akshay** **44 min** read

An organized collection of critiques and observations regarding the AFA's Youth Cyber Defense Competition coupled with thoroughly tested strategies to succeed.

Preface

Hi, my name is Akshay, known online as "hypernova." I want to acknowledge my privilege regarding the schools I attended, the supportive communities I was part of, and the comfortable financial resources I had at my disposal. I never had to worry about an absent coach or subpar computing equipment. I understand the inherent advantage I had, and I hope that the following content can help level the playing field for less fortunate competitors and their teams.

Introduction

I'm a three-time CyberPatriot Open Division National Champion and a CyberPatriot All-American Award recipient (given to competitors who qualify for nationals all four years of high school). Throughout my CyberPatriot career, I served as one of the Linux experts on my team and was the team captain of CyberAegis Tempest during my last season, CyberPatriot 15.

CyberPatriot XIII, XIV, and XV Open Division National Champions

As a competitor, I've witnessed many teams, including my own, devise various competition strategies that have failed to effectively and consistently produce results at a high level. In my last 2 years of competing my Linux partner, Safin and I approached the Linux portion of the competition with a method that enabled us to score the highest on the Linux images every single round for eight online qualification rounds straight, with our score margin increasing as the rounds became more difficult (evidenced below).

Competition Round	Our Linux Scores	Next Highest Linux Scores	Average Li
CP-XIV Semifinals	Debian: 100, Ubuntu: 98	Debian: 88 (-12) Ubuntu: 81 (-17)	Debian: 29
CP-XV Semifinals	Fedora: 62 Ubuntu: 76	Fedora: 35 (-27) Ubuntu: 53 (-23)	Fedora: 2.8

> Data for the statistics above only include platinum tier teams, sourced from CyberScores datasources

The pattern of low average Linux scores also extend to the Windows challenges for these past two seasons. Theses scores are disappointing, and I believe this is primarily due to major, widespread misconceptions regarding the nature of the competition. While my scores and strategies may not be perfect, I'd like to address some of these misconceptions, recount my own approach to the competition in detail, and suggest some improvements the competition organizers can make.

Misconception 1: What is CyberPatriot?

This might seem basic, but I've been approached with this question many times, and each time, I find myself at a loss for words. I've dedicated tens of thousands of hours to this competition, and I still find myself failing to describe it. Turns out, the way you define it can radically change your approach to the competition and how much you succeed.

Most competitors, coaches, and mentors will describe CyberPatriot as a Windows and Linux system-hardening competition, where, as the rounds progress, the vulnerabilities that must be fixed for points and advancement become increasingly obscure, esoteric, and even ridiculous. I believe this description is fundamentally inaccurate and wholly ignorant of the competition's structure and defining challenge. I do not necessarily blame anyone for the proliferation of this mindset; rather, I attribute it to the lack of clear messaging from the competition organizers themselves. While I do believe hardening is a component of the competition, it is not nearly the defining aspect. Here's how I would define the "images portion" of the competition:

CyberPatriot is a high school incident response and remediation competition. Teams step into the role of a corporate incident response team that has inherited a business computer network in the aftermath of a cyber-attack on the company's servers. During the competition, the teams work to uncover forensic data around the incident, reverse malicious or extraneous changes made by the threat actors, and further harden the system to protect it from future threats.

If one is familiar with the mechanics of the competition, I believe they will agree with my definition much more than the widely accepted one I've mentioned previously. If the challenge was simply composed of hardening, we would be given a blank/clean virtual machine to secure. Clearly, this is not the case. Based on the scenario given in every machine README, we can quite reasonably infer that we are given machines that have been broken into, implanted with various persistence measures, and contain items that break company policy. This thus characterizes the challenge as inherently response based.

Solution
Currently, CyberPatriot formally defines the "images portion" of the competition as the Network Security Master Challenge (NSMC). However, this term, which is a somewhat decent description, is not properly communicated as the name of the challenge at least until the national level, thus causing an unnecessary air of ambiguity around it. I would propose changing the name of the primary challenge to the Network Incident Response and Security Challenge (NIRSC), or something similar. This name emphasizes the corporate network scenario, incident response, and the hardening components of the challenge equally, bringing a strong and unambiguous definition to the challenge and offering a solid launch pad for new teams and coaches when determining their approach to the competition. However, simply renaming it is not enough; the challenge name should be unambiguously communicated to all teams. The competition organizers must refer to the images-portion of the competition as the NIRSC in all media, on the Challenges by Round website, and in competition emails.

Misconception 2: "Gotta Catch 'em all!"

Misconception two primarily stems from the previously mentioned and inaccurate idea that the competition is entirely centered around obscure or esoteric system settings. Due to the incorrect definition, students often fall into a dangerous habit of treating CyberPatriot like Pokémon.

Pokémon at a very basic level centers around the idea that Pokémon trainers collect as many Pokémon as possible (the show's main slogan is literally "Gotta Catch 'em all!") and lay them out in battle hoping to win.

> I understand this is a very simplified description of the franchise but it's just to serve as an easy analogy.

I believe that many competitors fall into the trap of trying to "catch" all the vulnerabilities they can during competition. They spend hours upon hours the weeks before a qualification round perusing through every system hardening guide they can get their hands on and adding every fix they find to their checklists or scripts. They then blindly throw these fixes at every image they see hoping that they get as many vulnerabilities as they can. They then write down these vulnerabilities and add them to their big "CyberPatriot Vulnerability Bank." While this approach to hardening is not completely wrong and even effective when used properly, it should not be the main strategy teams use when approaching a competition image.

Approaching the Competition

As much as I would like to just release my former team's scripts and checklists and get my team's hard work publicly appreciated, I will not be doing so. First of all, many of my former teammates are still competing and will be using those materials, and second of all I think that would just reinforce the harmful vulnerability collection mindset mentioned earlier. Instead, this section will be broken into descriptions of various skills, strategies, concepts, and values my team internalizes and employs to effectively and consistently score well on images.

Research (Where are all the vulnerabilities?)

Some say that CyberPatriot is a research competition, and that phrase is a good way to emphasize the importance research plays. The ability to research is a fundamental skill in computing and its adjacent fields, and is invaluable in a CyberPatriot context.

Research is critical before, during, and after each competition round.

In the weeks leading up to the competition, one of the primary components of your preparation should be research. Now, not all research is created equal. For example, if you're researching a critical service like Apache2 you shouldn't just google "Apache2 Security Settings," paste them into a checklist and move on. After you google these settings, you should be installing Apache2 on a clean image and configuring the directives you found, and restarting the service to ensure you're doing it correctly. This method offers a few advantages. Firstly, it will provide you with enough familiarity with the service so that, in most cases, it won't matter how the service is being used; you will be able to effectively configure it in the context of the scenario. Two, by setting it up yourself you'll notice possible security holes that are not mentioned in the popular guides on the first page of the internet. This ties into a couple of concepts I call "The Security Mindset" and "Respecting the Challenge" that I'll talk about later.

Research during the competition is also incredibly important. It's almost a guarantee that in every round you will encounter something you have never seen before. This could be a memory analysis technique required for a forensic, a really annoying apt error, or a critical service you've never seen before. Whatever it is, the clock is ticking, and you have a limited amount of time to figure things out; you'll need the ability to effectively use tools like Google, man pages, and Microsoft documentation on the fly.

Research after the round is important as well. If you really had trouble with a certain service or a README task, make sure you remember to go back a few days after the round and try to recreate the situation or something similar. Things do sometimes show up multiple times, especially services for prevalent protocols such as HTTP or DNS.

I know sometimes people have difficulty with what critical services to research beforehand, so here's a basic list. This list was constructed with an understanding of important internet protocols and technologies and their most popular implementations. The applications and protocols mentioned below should by no means be the extent of your research, but rather some things that you can start with.

</> Plaintext

```
HTTP/S: Apache2, IIS
DNS: Windows DNS, Bind9
Remote Management: OpenSSH, RDP
Database Technology: MySQL/MariaDB, MSSQL
Mail: Exchange, MailEnable, Postfix, Sendmail
File Sharing: FTP (vsFTPd, Filezilla, etc.), SMB (Samba, Windows SMB)
CMS: WordPress, Joomla
VPN: OpenVPN
Identity Management: Active Directory Domain Services
```

I'll cover researching operating system mechanics in "Operating System Familiarity."

Operating System Familiarity

At the end of the day, the competition is centered around operating systems, so a certain level of familiarity with the operating systems is necessary. A threat actor may embed persistence or cripple your system in a way where a fundamental understanding and familiarity with the relevant system's internals and normal function is necessary to proceed. Some examples include configuring Windows Registry in a way to deny all users PowerShell access or poisoning a Linux package to continually reinstall itself and cause a multitude of apt locking errors. Persistence measures such as DLL hijacking and malicious kernel modules also somewhat require an understanding of operating fundamentals to identify and remove.

The level of familiarity needed can be achieved in a few different ways. Some people completely immerse themselves in an OS, for example, dual-booting Ubuntu on their laptop and using it for their daily tasks. If you're into programming, you can develop applications on the operating system you want to learn about, for example creating a Windows app that utilizes the Windows API. You can also create practice images for others, learning both about how a system works and the possible security holes by attempting to add reasonable vulnerabilities themselves. Personally, I learned through the practice image route.

Operating system familiarity is also needed to effectively conduct security-related research. A casual Windows user may only know about the Settings app and Windows Defender, and that would be the limit of their research. An experienced Windows user with a solid understanding of its internals would know about the existence of utilities and services like Windows Firewall, Scheduled Tasks, and Local and Group policies. Thus, the extent of their research is wider and can cover more possible avenues to check for persistence and malicious misconfiguration or opportunities to further harden the system.

For now, I'll leave you with some basic system mechanisms to look into.

Linux:

- User and Password Management: /etc/passwd, /etc/group, /etc/shadow
- Pluggable Authentication Modules (PAM)
- Tunable Kernel Parameters (Sysctl)
- Package Managers (APT/DNF)
- CRON
- Permissions
- Systemd
- Sudo
- Polkit

Windows:

- Windows User Management
- Local Security Policy
- Group Policy
- Registry Editor
- Task Scheduler
- Windows Defender
- Permissions
- Service Manager

The Security Mindset

Alongside with being a long time Linux competitor I've also been responsible for teaching the 120+ students in CyberAegis Linux security for the past two years. The main philosophy I attempt to teach is something I call the "security mindset." That means having a solid understanding of security principles that can be applied to a multitude of scenarios. For example, let's say you understand the basic idea that logging is important because it allows us to understand what happened in the aftermath of an attack. Now, anytime you encounter a critical service or an authentication mechanism that has the ability to log, you will know to turn logging on and with a verbose setting, if possible. These are a range of principles that can be applied to services and other programs, such as hiding version information, enabling encrypted connections, and configuring secure permissions on sensitive items such as configuration files, files with password hashes, and raw database data directories.

To conclude, by having a toolbox of security principles, you can apply them to a wide range of possible scenarios you might find yourself in. It's especially useful when dealing with new services or technologies you've never seen before, hopefully grabbing you some vulnerabilities and an initial foothold on your situation.

> Additionally, it should be emphasized that learning about "red teaming" and "offensive security" components can greatly improve one's comprehension of ideas that blue teaming materials may tend to neglect or only briefly touch upon.

Respecting the Challenge

CyberPatriot is not real life, and so many people tend to take shortcuts when preparing and competing. For example, blindly copying and pasting directives into files, not knowing if those directives are deprecated or even formatted correctly, and not caring to restart the service afterward to ensure that their configurations were even applied.

Trying to do things the "right" or "intended" way is what I would define as respecting the challenge. This means:

1. Developing secure, modern configurations for services and ensuring that the service is restarted or reloaded so that your changes are applied.
2. Testing beforehand to make sure that your kernel configurations will not wreck your system.

3. Working on fixing errors before continuing rather than trying to force through them and just getting your hardening in.
4. And more.

I would be lying if I said my team doesn't occasionally take shortcuts, but for the overwhelming majority of our competition process we try to do things "by the book." This way of competing has helped us consistently avoid re-extractions and CCS penalties.

To Find Evil, You Must Know Good (Baselining and Meld)

Baselining is the single most consistent and effective strategy that I have employed throughout my time competing. Baselining is the idea of comparing clean or secured configurations and environments to the ones in the challenge image.

The first hour of our Linux strategy is spent comprehensively baselining. This means using third party tools and system package utilities to flag what files' default content has been changed. We scan for permission issues by comparing a list of world-writable files on a clean system to the one on the challenge image, flagging any differences. This can be extended to other possible permission issues like SUIDs. There are countless other parts of the system that can be baselined, such as default systemd units, scheduled jobs, processes, and more. By baselining in the first hour, I get a pretty good understanding of what a majority of the vulns on the image might be.

Another powerful tool our team used was Meld, which is a visually based diffing and merging tool for files and directories that works on both Linux and Windows. To effectively use Meld, my team meticulously incorporated entire default filesystems into our script, and enhanced its security by implementing our carefully tailored, secure configurations. During competitions, all of our config hardening was done through Meld.

While not as easily applicable, baselining can be used effectively in a Windows context as well, such as examining configured GPOs on the challenge image or comparing default Windows folder permissions with those harvested during competition.

Automation (Consistency vs Speed)

A large component of our qualification round strategy was the use of robust, heavily tested scripts. Our scripts are not merely just PowerShell or Bash commands thrown in a file, but somewhat complex frameworks built with an emphasis on consistency and extensibility. Our Windows script is known as Solaris and was started in CyberPatriot 13, the first year we won Nationals, with 650 commits and counting. Our Linux script is affectionately called Atlas, and was also started in CyberPatriot 13; today it has over 1000 commits.

When the scripts were conceptualized, our team's primary goal was not speed but rather improving consistency. At the time, we were heavily focused on making sure we qualified for nationals, as the newly instated two-team rule would essentially make it so that we had to place in the top two at semifinals to qualify. We were pretty confident that, with the combination of our somewhat unique strategies and comprehensive research, we would not have trouble scoring more "difficult" or "obscure" vulnerabilities. Our main concern was consistently scoring the "easy" points that we believed would be the distinction between qualifying or not. That meant putting extra time into completely scripting items like users, including parsing the README to minimize human error, and developing a method that would consistently flag all manually installed packages on the system.

The overvaluation of "extra time" is a trap I often see competitors fall into when building their scripts. They tend to build fast scripts rather than ones that focus on consistency or comprehensiveness so they're afforded more time left to hunt for the most challenging vulnerabilities. There's two major problems with this approach:

1. They tend to allocate all their prep time to improving their script speed by a couple minutes. And thus, they lose sight of actually improving the amount of content they know. Sure, you can add and delete users + harden a couple of configuration files in 60 seconds, but if that's the extent of your knowledge then you haven't really accomplished much
2. People tend to not have a plan to find vulnerabilities they don't know about after their script finishes, so they'll sit around for hours clicking around and getting passed on the scoreboard.

So, what are my recommendations for approaching scripting?

- Focus on what you can script consistently instead of scripting everything.
- Put effort into developing a robust script, make sure it's easily extensible and heavily tested under a wide range of scenarios.

- Develop a plan on how to find vulnerabilities that you don't know about, that means using methods like baselining in the beginning of competition to flag possible vulnerabilities.

Testing and a "Gameplan"

It goes without saying that you should be constantly testing your new additions to your scripts and checklists, developing a solid understanding of how to implement your security fixes and what fixes may make the system more delicate. It is unfortunately not a rare occurrence to see teams have to re-extract multiple times during a competition round due to their script breaking something.

You should be testing your script heavily and robustly, putting it through a range of scenarios. Through your practice, you should also be developing your competition "gameplan," which makes sure there is a set order to how you do things. That means not configuring your, possibly dangerous Sysctl configurations until after you restart for kernel updates. Play smart.

Here's a high-level overview of my Linux "gameplan." The detailed version of this includes which modules of our script to run with each section and any other manual work that needs to be done.

1. README
2. Forensic Questions
3. Initialize script and fix potential issues that may affect script features (e.g., reset APT repositories, removed immutable permissions, reset $PATH variable, etc.)
4. Baselining
5. Users & Groups
6. Backdoor & Malware hunting
7. Updates
8. Reboot to apply kernel updates
9. Package and service management
10. System hardening (PAM, Sysctl, Sudoers, Polkit, Bootloader, etc.)
11. Critical Service security
12. File and directory permissions

Team Composition

As CyberPatriot is a team competition, the team composition is paramount for achieving success. As the competition becomes increasingly complex and less formulaic, building a well-rounded team is essential to excel.

I have personally witnessed heart-wrenching moments when teams of five, comprised of exceptionally talented and driven competitors, crumbled under the immense pressure of trying to handle every aspect of the competition. These teams, despite their potential, often fell short of reaching their true capabilities in the competition. A prominent illustration of this occurred with GJD^3 from Colorado during Cyberpatriot 15, who, in my belief, narrowly missed third place at the National Finals primarily due to their overreliance on one member. This team member had to juggle the critical roles of being the main Cisco competitor while simultaneously holding a pivotal position in the NSMC.

The circumstances surrounding GJD^3 serve as a reminder of the value of team composition and the dangers of placing an excessive amount of reliance on individual abilities. When juggling several important obligations, even the most outstanding contender would likely feel overburdened.

CyberPatriot XIV and XV Open Division National Finalists

Thus, I firmly believe all teams should have a full roster of six members whenever possible. While the current rules only allow five members to compete at a time, having six members offers the advantages of greater manpower and a larger permutation of available skills and expertise. A team of six can simply research more than a team of five, maximizing preparation during the short turnarounds between competition rounds. Additionally, adding an additional team member enables competitors to specialize more. Each participant can concentrate on developing their abilities in particular competitive categories. For example, a devoted Cisco competitor can focus entirely on honing their talents in Cisco-related activities without worrying about other challenge elements. This specialization not only increases team performance as a whole but also increases individual proficiency. The substitution rule can then be used to a team's advantage, by allowing a different skillset to be introduced sometime during the competition round to address the specificities of the round.

> Please be advised that the substitution of a competitor is limited to a single occurrence within each round. Once this substitution takes place, further replacements are not allowed.

The other aspect of good team composition is an effective distribution of specializations. I believe the following team composition would be the most optimal for the online competition rounds:

- 2 dedicated Windows competitors
    - Windows consistently emerges as the most time-consuming and demanding component of the competition (Excluding CyberPatriot 15 Fedora).
        1. The first three qualification rounds, teams encounter two Windows images in comparison to a single Linux image.
        2. Certain intricacies commonly related to general Windows security mostly prove to be less amenable to automation than Linux machines.
        3. The Windows challenges tend to be more plagued with issues (broken PowerShell, disabled access to system mechanisms, etc.) that require more time to analyze and rectify.
- 1 dedicated Linux competitor
    - While Linux is also a demanding aspect of competition rounds it tends to be more manageable by a single competitor than Windows. Despite its challenges, I consider it to be more tractable for individual competitors.
        1. As previously mentioned, Linux plays a smaller role in the first three qualification rounds.
        2. Due to its configuration file-based and command-centered structure, Linux lends itself more readily to automation compared to Windows machines which makes it more manageable in competition rounds.
- 1 Linux/Extra Challenge competitor

  - The responsibility of this competitor is to provide assistance to the primary Linux competitor. They will complete Linux-related tasks directed by the primary Linux competitor, this would likely include Forensic Questions, troubleshooting, or completing certain checklist sections on the challenge images.
  - The other responsibility of this competitor is to pursue points in any extra challenges that may be included in the competition round. It is important that this competitor is well-versed in a variety of cybersecurity topics and technologies as the Extra challenges often encompass a diverse range of skills and scenarios.
- 1 dedicated Cisco competitor
  - This competitor should be exclusively focused on Cisco throughout the competition season. Cisco is an integral component of the competition that can be easily mastered with dedication, resulting in consistently high results that optimize the team's overall score.
- 1 "Jack of All Trades" competitor (likely substitute)
  - The "Jack of All Trades," is a versatile competitor with a skillset that spans all facets of the CyberPatriot competition. While not needing to be a master in any specific area, this individual serves as an invaluable asset, providing essential assistance at critical junctures and acting as a valuable sanity-checker for the team.
    1. For example, since Cisco is a competition component which demands precision without live feedback, another competitor with a Cisco background reviewing the primary Cisco individual's work is immensely valuable and can eliminate or minimize any possible minor mistakes which could have significant consequences.
    2. Another advantages of this competitor include their fresh perspectives, solutions, and/or assistance. This could include identifying workarounds for a broken system mechanism on the Linux machine or offering to assist with more mundane, yet important tasks, allowing specialized competitors to focus on more complex matters.

Team Cohesion

This is a more generalized piece of advice as team cohesion is integral in any team-based competition. It would seem obvious that establishing strong teamwork skills and good communication is integral in any team-based competition. However, its significance, at least in

CyberPatriot, cannot be overstated, as it forms a solid foundation for success. I do understand that this is not completely possible for all teams as, unfortunately, absent or uninterested teammates are quite common. There are also examples of teams with a somewhat disjointed and individualistic approach to competing that have been incredibly successful, such as Team Half Dome. However, I firmly believe that strong teamwork and a cohesive team are invaluable, those values have always allowed my team to consistently push forward for so many years, despite the challenges we faced throughout the competition. To achieve this, here are a few ideals worth striving for:

CyberPatriot XIV and XV Open Division National Runners-up

Regular meeting times

Scheduling regular team meetings is essential to keep everyone on the same page and building friendships within the group. Consistent, weekly 2–3-hour team meetings were the lifeblood of our team. Anyone in our team will fondly look back on solving bugs in our scripts or figuring out how to set up a critical service together while eating pizza. It is these team meetings that built friendships and a sense of camaraderie that took us as far as we could possibly go.

Consistently creating, assigning, and completing tasks

It is important to divide up the work between the team members, but to do it in a public way as to emphasize transparency and accountability. A system like a shared to-do board allows any teammate to identify who is currently working on related tasks or who might have relevant expertise. Moreover, the shared to-do board offers transparency into the team's current priorities. As tasks are added, updated, or completed, the board provides real-time visibility into the team's progress. This helps align everyone's efforts towards achieving common goals and ensures that team members are on the same page. In terms of accountability, a public to-do or scrum board can hold team members responsible for their work. With tasks publicly displayed, there is an inherently higher sense of ownership and commitment attached to them. It also allows the team captain or other team members to identify "bottlenecks," allowing for an easy avenue to either confront "slacking" teammates or help them with tasks they are finding difficult. Our team had a very simple, custom, to-do Discord bot on our server that would hold a list of things we had to do and who they were assigned to. Tasks ranged anything from “Test new changes to Linux script” to “start MSSQL research.” While building a custom to-do bot is unnecessary, it’s easy to find a free online scrum board or even a Google Spreadsheet to manage tasks and track progress.

Shared and Easily Accessible Resources

In fostering a truly cohesive and effective team, it’s important that all resources created by individual team members are made readily available to everyone else within the team. Unfortunately, it’s not uncommon to encounter teams where each individual develops separate checklists or withholds their scripts from the rest of the team. However, this approach proves detrimental to the overall camaraderie and trust that underpins a successful team dynamic. To share and develop resources together, I suggest a shared, well-organized, Google Drive to store and edit checklists and creating a GitHub Organization for scripts or other automation.

Why did I just tell you all of that?

In the process of editing this post, I had someone provide me this critique: “How would you feel if you spent 3 years understanding how the competition works and developing strategies that consistently work just for some dude to spoon-feed them to everyone, good strategies give you a significant edge over other teams and I think that edge should be learned and earned, not given.” and I’d like to respond here:

First of all, my team, the developers of the above strategies, finds no problem with this post and its contents and some of them even encouraged me to add even more. While we do understand that it is not our job to teach other competitors, we also love the competition and want to see it and its community succeed. Public projects and resources like Elysium, Linux, Visually, King Arthur’s Castle, and Operation Titan Shield were all initiated by our very own team over the past three years. We don’t just want to do well in this competition; we want to thrive in a competition that is also thriving.

Second of all, almost none of the strategies I mentioned above are effective without a deeply strong conviction to succeed and a breadth of time at your disposal. For example, baselining is a tedious strategy and takes hours and hours of testing and discipline to pull off correctly every round. Even regular meeting times, as simple as it sounds, requires a relatively strong level of commitment. During rounds, respecting the challenge is a value that is difficult to adhere to, especially under a time crunch. The bottom line is that if you put in the time, you will succeed. There are no shortcuts.

Zero to Hero

I’ve dumped a lot of information and suggestions throughout this post, and I understand it may be overwhelming and difficult to figure out where to start. Below, I’ll try my best to consolidate my advice into a small list of somewhat broad, actionable steps to take a new CyberPatriot team all the way to the National Finals.

> I have framed the following advice through the lens of an Open Division team; however, it can easily be applied to the All Service division.

1. Lay the Groundwork:
    1. Start as early as possible; the best teams work through the summer to prepare for the upcoming season. I understand that this might not work for everyone, as the school year typically doesn’t start until mid-August at the earliest. However, to compete against the best of the best, who have been competing for years, it is best to start as early as possible to have a chance.
    2. Do your best to assemble a well-balanced team, ideally attempting to follow the example arrangement detailed in “Team Composition”.
    3. Forge a cohesive team by setting regular meeting times, utilizing task management tools, and ensuring infrastructure for the future placement of team resources and tools.
    4. Do your best to get registered early for the competition and establish robust and easily accessible lines of communication with your coach, as you will need them ALL the time.
2. Experience the Challenge:
    1. The AFA releases exhibition and practice rounds through the spring, summer, and early fall before the competition formally begins. These images tend to be very easy, easier than the images you will encounter in the first competition round, and some of them are shipped with answer keys. Despite the lack of difficulty, they are a good introduction to the competition environment.

        1. There are also some public, very easy practice images that can also be a good introduction to the competition:
            1. Bloons TD 6 Server (Debian 10) [Author: *mobmaker*]
    2. You can also choose to experience how difficult the competition can become through publicly available, community-built practice images. Here are a couple I recommend if you would like to get a glance how difficult the competition can become:
        1. Mushroom Kingdom Server (Windows Server 2019) [Author: *Magistrate*]
        2. Space Force Server (Debian 8) [Author: *Magistrate*]
3. Join the Community
    1. You are not alone, there are thousands of competitors across the nation with the same aspirations as you. Together they've built an expansive, inviting, and robust community that provides anyone access to practice material, resources, and other successful competitors. Join on Discord: https://discord.gg/cyberpatriot

    > I would highly recommend following the "announcements" channel in the discord to easily stay informed regarding competition changes and updates.

4. Develop OS Familiarity and Start Research
    1. Look back at the Operating System Familiarity section and begin learning about and figuring out how to secure different components of the operating system you'll be specializing in. Record your findings and share them with your team along the way.

    > A helpful resource for Linux is a YouTube series called "Linux, Visually" which is currently under development by my former Linux partner, Safin Singh. It does an excellent job of explaining Linux mechanics in a way that is easy to understand. You can also interact with Safin himself in his Linux, Visually Discord Server.

    > There are also many TryHackMe rooms that cover fundamentals of many computing and OS concepts. Some useful ones are Windows Fundamentals 1-3 and Linux Fundamentals 1-3. There are also various rooms that cover other topics such as Active Directory, explore on your own.

> Another resource that may help in understanding and experimenting with Linux components and other technologies is the Mr. Robot Practice Image (Fedora) [Author: *tirefire*]

> To gain familiarity with the Linux terminal and Powershell you can attempt OverTheWire Bandit and UnderTheWire Century challenges.

2. Attempt to apply the concepts discussed in the Research section to widen and deepen your understanding of Linux and various critical services. Be as hands-on as possible to build up your skills and be able to effectively apply them not only in competition, but eventually in the real-world as well.

5. Researching security
    1. There are several resources out there to quickly gain some OS security knowledge and strategies. However, they should be used effectively. For example, there are several robust OS hardening scripts on GitHub, but when borrowing content from them, first attempt to understand it and then test it before adding it to your own checklists and scripts. Below are a few links to some general resources to get you started.
        1. Operating System Security Standards (STIG, CIS)
            1. STIGS: https://www.stigviewer.com/stigs, look through these for relevant ones such as the Ubuntu 20.04 standard and Windows Server 2019 standard.
            2. Download up-to-date CIS benchmarks here, you can also find some older ones on Github repositories such as this one.
        2. GitHub OS Hardening/Vulnerability Scanning Scripts
            1. Konstruktoid: Hardening Ubuntu, Systemd edition.
            2. HardeningKitty and Windows Hardening Configurations
            3. PEASS - Privilege Escalation Awesome Scripts SUITE (with colors)
6. Develop a Plan
    1. After conducting foundational research it’s time for the team to unite and strategize an approach the competition challenges. While determining competition priorities, the team should be developing preliminary checklists and identifying tasks that can be reliably automated.

7. Practice and Address issues
    1. As a team, practice with publicly available practice images and test your checklists and scripts as if you were actually competing. Here are a few images I suggest, they are "medium" difficulty.
        1. "Money Heist" Practice Image (Ubuntu) [Author: *KaliPatriot*]
        2. "rooReaper Strikes Back" Server (Ubuntu 22.04) [Author: e*th007*]
        3. Windows Practice Image (Windows 10) [Author: *KaliPatriot*]
        4. "The Cincinnati Zoo" Practice Image (Windows 10) [Author: d0nkeyman]
        5. Santa's Workshop Laptop (Windows Server) [Author: *Magistrate*]

        > Keep in mind that these images may not reflect the actual operating system versions used in the competition.

8. **Ideally all of the above steps will be completed before competing in Qualification Round 1.**
9. Compete
    1. If your goal is to go to the National Finals you should aim to full score each image in the first qualification round, however, as the qualification rounds get harder every year it is acceptable to miss one or two vulnerabilities total, across all images. Your networking score should be very high as well.
    2. Since the first qualification round is very simple, it's usually not a good predictor of semifinals success. However, teams with aspirations of qualifying for the finals should aim for a top 30 or 40 finish.
    3. Throughout the competition the team captain should take notes of issues that pop up to review at the subsequent team meeting.
10. Improve and Continue Preparation
    1. Your first experience in the competition environment may reveal significant organizational issues within the team and will require immediate resolution as the next round looms just a few short weeks away. However, the first qualification round should not expose any knowledge shortcomings, as the challenges have historically been and are expected to be relatively simple.

   2. Most of the top teams adopt a mindset of not preparing for Qualification Round Two at this point, but rather for State Round and Semifinals (Qualification Rounds 3 and 4, respectively), the competition rounds that are notoriously difficult and contain the highest concentration of “curveballs.” If there were no significant knowledge gaps identified (missing more than 1 or 2 vulnerabilities total) during Qualification Round 1, then this is the mindset your team should adopt as well.

11. Compete again (Qualification Round 2)
    1. Qualification Round 2 may be where you encounter knowledge gaps for the first time as the complexity of the images increases. Ideally, you should strive to miss no more than 4 to 5 vulnerabilities total across all images, score high on Cisco, and secure a spot among the top 25 teams.

    > As the competition progresses, the rounds serve as increasingly accurate predictors of success at the semifinals. A common observation is that national finalist teams display a remarkable consistency, either maintaining their top positions or steadily ascending the ranks with each subsequent round. The top teams’ comprehensive knowledge base and inherent intuition in regard to cybersecurity and incident response scenarios enable them to achieve that consistency despite the increasingly varied and complex scenarios handed to them.

12. Repeat Step 9
    1. While you may uncover a few more organizational challenges that should be easily fixed, the bigger challenge will be strategizing on how to deal with the increasingly complex critical services and scenarios inevitable in the future competition rounds.
    2. It is paramount to have developed a solid plan before the state round, which is infamous for its significant difficulty spike. With the images already posing a significant challenge, any organizational, checklist, or “gameplan” inconsistencies can exacerbate into a substantial failure during competition. I believe the time between Qualification Round 2 and State Round can make or break teams’ chances for Nationals qualification.
        1. To effectively navigate this crucial stage, adequate planning, teamwork, and meticulous attention to detail become **essential**. Make sure that everyone on the team is on the same page and that any errors from the previous round have been fixed. Both critical

service research and OS research need to be kicked into overdrive to effectively tackle the upcoming challenges. By this point, all scripts or other automation techniques should be easily extensible and consistent.

3. Keep practicing. Here are a couple more public practice images that either match the difficulty level or are slightly easier than those you'll encounter at State Round:
    1. Among The Reindeer (Windows 10) [Author: *Magistrate*]
    2. friends of the fellowship Windows Server (Windows Server 2022) [Author: *firepony57*]
    3. Narnia Minecraft Server (Ubuntu) [Author: *eth007*]

    > At State Round, you'll likely encounter "broken" OS components such as a non-functioning PowerShell or misconfigured environment variables. To practice fixing "broken" images, I suggest the following image: My Little Pony Practice Image (Windows 10) [Author: *d0nkeyman*]

13. Compete (State Round/Qualification Round 3)
    1. State Round will likely expose significant knowledge gaps, with most of the top teams missing an average of 7 to 8 vulnerabilities across each image. The leaderboard will likely look very different from the much easier rounds 1 and 2, providing a mostly accurate depiction of what the Semifinals results will look like. If your goal is to qualify for the National Finals, you should be placing in the top 12-15 teams in this round.
14. Prepare
    1. By this stage, if your team is genuinely capable of progressing to the Nationals, you should have developed a high level of self-sufficiency, enabling your team to wisely determine the required areas of focus and specific tasks needed to be accomplished before the final qualification round. If you aspire to compete at the National Finals, a world-class experience alongside the most motivated, driven, and intelligent high-school students in the nation, leaving no stone unturned becomes imperative. Leave as little to chance as possible and harness every opportunity to improve and tighten your strategies through thorough preparation.
    2. As always, keep practicing. Here are a couple more public practice images that either match the difficulty level or are slightly easier than those you'll encounter at the Semifinal Round:

1. Princeton-Plainsboro Teaching Hospital (Windows 11 Education) [Author: *x1nni*]
2. King Arthur's Castle (Windows Server 2019) [Author: *altoid0*]
3. Mushroom Kingdom Server (Windows Server 2019) [Author: *Magistrate*]
4. friends of the fellowship Linux Server (Fedora) [Author: *eth007*]
5. Space Force Server (Debian 8) [Author: *Magistrate*]

15. **Compete, qualify for the National Finals and begin your legacy.**

Competition Improvements

Competing in CyberPatriot and being a leader at my school's organization, CyberAegis, has been the single most formative experience of my high school life. It's opened countless doors for me and ignited passion that I'll be pursuing for the foreseeable future.

Unfortunately, CyberPatriot's reputation is not always positive, as it is perceived by many competitors as a recondite and unrealistic challenge with limited opportunities for genuine learning and growth. Although I disagree with this commonly held view of the primary challenge, I do understand other valid frustrations about the competition culture and the dearth of opportunities to improve. Here are a couple of issues I've noticed within CyberPatriot and its culture, and some potential solutions that the organizers can undertake to address them effectively.

Eliminate unnecessary ambiguity around the primary challenge.

As previously mentioned in "Misconception 1: What is CyberPatriot?" I believe the "Images Portion" of the competition, formally known as the Network Security Master Challenge (NSMC) should be renamed to the Network Incident Response and Security Challenge (NIRSC) to more accurately describe the nature of the challenge. Moreover, the competition organizers should proactively advocate for the adoption of this revised name to ensure that the primary challenge is consistently and accurately referred to as the NIRSC. This cohesive, unambiguous terminology will facilitate a better understanding of the challenge's objectives and contribute to a more coherent and precise communication across all competitors, coaches, and even potential sponsors.

Encourage and reward community initiatives.

CyberPatriot largely overlooks the recognition and the rewarding of community initiatives developed by competitors and former competitors. While they may, understandably, avoid endorsing entire platforms such as the CyberPatriot Discord server or remain cautious about assuming any responsibility for potential consequences related to endorsing community projects, there remains an expansive pool of public resources generated by individuals that go unnoticed by many competitors and coaches.

However, CyberPatriot has acknowledged unofficial resources in the past such as Cyber Replay as a likely scam, without assuming any responsibility for potential associated consequences. Therefore, the contrapositive must also hold true: CybePatriot could acknowledge more legitimate resources such as the Elysium Suite or the Magistrate Project, and still maintain zero responsibility or control over the other products.

CyberPatriot's current stance minimizes positive reinforcement and public recognition that incentivizes further public resource development. To remedy this, I would suggest including a "Community Section" in the CyberSentinel or the official website that would consolidate useful public resources like the Elysium Defensive Cybersecurity Training Suite or the various practice images developed by competitors. Doing this would create a positive feedback loop for the CyberPatriot community with a dynamic, ever-evolving ecosystem of community resources, and ultimately uplift the entire community and make entrance into the competition less intimidating. This minimal effort will also better align CyberPatriot to its core mission of inspiring K-12 students toward careers in cybersecurity or other science, technology, engineering, and mathematics (STEM) disciplines critical to our nation's future.

Further Embrace the Incident Response Scenario for Competition Challenges

While the competition images have shown significant improvement over the years, there is still room for enhancing their alignment with the Incident Response (IR) scenario they aim to simulate. Presently, the aftermath of the attack is depicted through various persistence measures and unauthorized items scattered throughout the image. However, these elements often lack a cohesive connection to a central attack on the machine. Moreover, the Forensic Questions occasionally miss the mark by focusing on unrelated aspects such as complex Regexes or hidden messages in files, rather than directly addressing the actual attack orchestrated by the scenario's threat actors.

A valuable step towards enriching the competition experience would be to center more forensic questions around the specific attack scenarios portrayed in the images. For instance, even in a round 1 image, a simulated brute force attack on SSH could be incorporated, leading to forensic questions like “Which user did the attacker target through brute force, and how many attempts did it take for them to gain unauthorized access based on authentication logs?” Similarly, in a more advanced scenario like the semifinals, obtaining Remote Code Execution (RCE) from a vulnerable plugin installed on a WordPress site, and then escalating privileges through a required cronjob, could inspire relevant questions pertaining to the exploit and its impact.

By fine-tuning the competition images to closely align with incident response scenarios and tailoring forensic questions accordingly, CyberPatriot can create a more immersive and realistic experience for participants. This would further enhance the applicability of the competition to the real world and further align CyberPatriot with its own goals.

Closing Thoughts

This post was initially a concise description of my competition strategies with the purpose of helping other competitors develop their skills and make success more attainable. However, as I delved deeper into my methodology, I believed it was important to address misconceptions regarding the competition, suggest improvements, and attempt to help foster the larger CyberPatriot community. I truly believe that CyberPatriot is a unique program that has garnered great success, has untapped potential, and can better achieve its goal of cultivating the next generation of cybersecurity professionals with a few key improvements.

I understand that I was very privileged to start competing in CyberPatriot in Middle School and with, arguably, the most successful organization in the competition’s history. Despite having had this privilege, I believe I at least somewhat understand the significant challenges that students face when clawing their path to the national finals. I am deeply inspired by teams who climb their way to the national finals without an already established successful program, such as KaliPatriot, Cyber Wolves II, Half Dome, c¥b3rh0u#d5, Lone Star Company NJROTC + Moey, and many more. I believe their stories and accomplishments are some of the most impressive and meaningful in the history of the competition; they exemplify that ingenuity, passion, an effective strategy, and plain hard work can turn underdogs into champions.

This post is my attempt at leaving behind a positive impact on the CyberPatriot community and competition, hopefully helping new or struggling teams claw a path to the National Finals. I'd also like to thank Tanay, Safin, Raadwan, Tyler, Brycen, and Nicole for looking over and helping edit this post. If you have questions about anything I mentioned in this post, please don't hesitate to reach out on Discord (handle: hyper.nova). Thanks for reading.

posts

CyberPatriot, Advice

Share:

OLDER	NEWER
-	-