Open Source and Commercial Capture The Flag
Cyber Security Learning Platforms: A Case Study

Matthew Swann*, Joseph Rose*, Gueltoum Bendiab*, Stavros Shiaeles*, Fudong Li*
*Cyber Security Research Group, University of Portsmouth, PO1 2UP, Portsmouth, UK
Matthew.Swann@myport.ac.uk, Joseph.Rose@myport.ac.uk, gueltoum.bendiab@port.ac.uk,
sshiaeles@ieee.org, fudong.li@port.ac.uk

Abstract—The use of gamified learning platforms as a method of introducing cyber security education, training and awareness has risen in the last years. With this rise, the availability of platforms to create and host the challenges has also increased. In order to identify their suitability for cybersecurity training we need a method to compare them using some features. In this paper, we are trying to identify what features are needed for a Capture The Flag (CTF) platform through literature review, research opensource and commercial CTF platforms available and compare them against these features in order to conclude on the best platforms available for cybersecurity training.

Index Terms—Security, Capture the Flag Competitions, Gamified Learning Platforms, Cyber Security

I. INTRODUCTION

Over the last decade, the cyber security learning platform landscape has changed dramatically, with a greater emphasis on shifting technical and applied cyber security practice learning from a theory-first to a practical approach. The proliferation of capture the flag platforms has been a critical component of this evolution. A capture the flag challenge entails developing and deploying various challenges centred on different aspects of cyber security, such as binary exploitation, web security, and reverse engineering. Each of these challenges can be exploited in one or more ways, with the end goal being to solve the challenge and obtain the flag, a string of text, which can then be submitted for points. There are many different versions of these platforms that allow for the hosting and delivery of these competitions and can meet the needs of various organisers and the objectives they wish to achieve. In this section, we will compare a number of popular platforms that are used for these purposes in order to determine the best use cases for each platform.

This comparison will be conducted by identifying the key features that are needed for an ideal platform, by comparing what attributes that the most popular self-hosted and commercial offerings provide. We can collate these features and use these as a metric for identifying the best platforms.

We will then apply these metrics to a selection of the most popular platforms to identify the best self-hosted and commercial platforms. We then posit that an ideal gamified learning platform would apply all of the identified features, and how that compares to current offerings. Using this information, we will then identify areas for future work and what next-generation platforms should provide.

II. RELATED WORK

In Adams and Makramalla’s work [1] the authors describe an ideal gamified learning approach to cyber security as including four main gamification elements in order to promote engagement and allow practical learning of cyber security skills and practices. These are: progress mechanics, player control, problem solving and story. They propose that these attributes, when combined with practical challenges focused on specific aspects would allow for an ideal platform, from a learning standpoint. We can take these attributes and apply them to our comparisons through the factor of how and why tasks are grouped, feedback on player score and progressions and ease of player interaction through built-in interactive environments.

The work of Gonzalez et al [2] proposes the use of varied challenges, completing using a capture the flag platform, as the basis of a syllabus on practical cyber security. In part of this work, the authors break down what attributes of different platforms are imperative in order to successfully utilise the platforms in this way. They note that the exploitable resources should be available locally, as downloadable files or remotely accessible through cloud deployment or deployment through the platform itself. Furthermore, emphasis is placed on the importance of the virtualised, or containerised nature of these systems, specifying that no specialised equipment should be needed to deploy these activities. We can apply these practical requirements to the platforms we compare, evaluating the simplicity and ease of deployment for the exploitable machines and challenges provided by each solution.

In a work by Malone, et al [3] the authors propose a gamified practical cyber security learning platform in a similar style to other capture the flag frameworks, in their research they stress that the modularity and flexibility of the framework in order to provide customisable and adjustable learning scenarios as well as extensibility for the end user to create and adapt the platform to their own needs. We can build this viewpoint into our comparisons by assessing to what extent extensibility is factored into the design and implementation of each platform, through the support for API systems or other end-user customisability or configuration.
A study by Ros et al [4] examines the importance of feedback to the player in gamified cyber security learning platforms. The researchers noted that aside from the design of the game and learning material- the most important aspect of user feedback was being able to see the reward, in the form of points awarded, per task completed. We can apply this to our assessment of each platform through observing how the scoreboard is displayed, added to as well as what filtering options to refine the scoreboard by team, region or associated institute to make viewing progress for the end user easier.

It is worth noting that there is very little contemporary literature comparing the current platforms that are available based on their feature set specifically, as we will do in this study.

### III. Types of Challenges

Before we can compare specific platforms, we must first categorise them according to the type of challenge format they are attempting to offer. There are two kinds of capture the flag challenges. The first and most common type is Jeopardy-style challenges, which provide multiple challenges that each team can see and access. The difficulty of these challenges varies, and the points awarded for completing the challenge and then submitting the correct flag increase as the difficulty increases. There are bonus points available for being the first to solve a problem, which are referred to as "first bloods." These points are added to the team’s total score, and whoever accumulates the most points over the course of the competition, which is typically 24 to 48 hours, wins. The second, less common form of challenge is known as an Attack-Defense challenge. These are not like Jeopardy-style challenges and do not have as rigid a framework. Every team will typically have their own virtualised instance of the challenge machine. They must protect their instance while also hacking into the instances of the opposing team. There are a few popular variants of this form of challenge, such as King of the Hill[5], where there is only one instance and every player attacks and defends on the same instance; or Battlegrounds[6], where each player on each team has their own computer to defend while also attacking, raising the difficulty.

### IV. Feature Comparison

By reviewing the current offerings for interactive, gamified cyber security training platforms, we are able to assess the merits and drawbacks of each platform. By doing so, we noticed an interesting pattern that can inform the optimal feature set for an ideal gamified cyber security training platform.

#### A. User Experience and Workflows

It is evident that each platform, whether it be a self-hosted framework or a commercial offering, has to provide several key features in order to satisfy the workflows of users and organisers. Firstly, there must be the ability to set and access tasks. These tasks should be logically grouped, in the observed solutions these were under names like "rooms", "machines" or "challenges". These should contain the task material. If the material is an interactive exploitable machine, then an in-browser interactive shell, like TryHackMe or PicoCTF should allow this; otherwise a per-user VPN connection to the machine should be implemented. The user should be able to submit and recognise that the flag they have submitted is correct and if so, how many points have been rewarded for doing so.

#### B. Coordinator Features

The platform should allow for easy administration of users and teams as well as construction of tasks. This process should be easy to implement and allow for uploading and deployment of machines natively. Furthermore, cutoffs for flag submission and machine deployment should be easy to set and adjust on a per competition or per task basis.

#### C. Scoring

The scoring system for the platform should be configurable for the end user, able to adjust the points awarded per challenge- in order to scale the reward for completing harder challenges. Furthermore in Attack-Defence challenges, implementing a dynamic scoring system, such as the Elo inspired system used by ForcAD should be available as an option for the event coordinators to configure. Alternatively, a system similar to Microsoft’s TrueSkill rating system, which is calculated based on Bayesian inference of player skill and could be used for the same purpose. The scores, for both individuals and any associated teams should be visible. Usually, this is accomplished using separate scoreboards. These scoreboards should be refined further through national or institution, e.g. associated University, filtering.

#### D. Deployment

Deployment of the platform, as well as set tasks should focus around portable deployment, such as virtual machines and containerised solutions such as Docker or LXD. In addition to portability, this is in order to allow customisability and ease of setup for organisers.

#### E. Extensibility and API Integration

The platform should provide API integration, ideally in multiple methods such as RESTful and SOAP. This is to allow for end user and community driven extensibility. This is a key feature that is not present in most currently available solutions, and would provide an excellent area of expansion for future platforms. Examples of this extensibility could include customisable web scoreboards or graphics for specific events, teams or user actions.

### V. Jeopardy Style Platforms

#### A. OpenCTF

The OpenCTF[7] platform is a PHP-based, jeopardy-style CTF platform with a web-based front-end and a scoreboard, as well as easy instruction-based, hyperlinked, or file-download based challenges. This configuration’s no-frills architecture and functionality offers several advantages. For instance, it
is extremely lightweight and simple to use, usually running on a single NPM stack. The fact that problems can be generated by simply editing the appropriate database table. This enables effective exploitation challenge machines to be set up on the same server, possibly utilising containerisation. This facilitates an incredibly powerful, scalable, and simple-to-deploy challenge framework. However, its advantages may also be its disadvantages; it has a rather simple user interface and can seem dated as compared to other platforms. It also lacks some of the features provided by other sites, such as team monitoring and live popup score notifications. There are several other variants on this framework, each of which builds on top of OpenCTF’s core features, such as Christmas-CTF[8], which provides both thematic improvements as well as UI updates and functionality additions.

B. TinyCTF

TinyCTF[9] is one of the most widely used CTF networks, with extensive use in many large and small-scale CTF competitions, including NahamCon’s CTF[? ] TinyCTF is made up of Python and Javascript components, and it utilizes Python’s Flask module as a web server—though the documentation suggests running flask on top of a proper web server like Nginx or Apache if a large number of players are anticipated. TinyCTF allows configuration of tasks in a JSON file that are then imported using a built-in python script; this allows for easier formatting and configuration of tasks that have multiple parts, or multiple flags. Notably, it also contains compatibility for easy deployment on EC2 instances, which make it extremely easy to build, configure and deploy the CTF platform in a scaleable way. There is also the option of installing the platform manually on multiple Linux distributions. A major advantage of TinyCTF however, is the installation wiki [7] that covers manual installation step-by-step, making installation and configuration easier. It is a light to medium-weight platform, that features a clean and intuitive UI, but it is not as minimal as the previously discussed OpenCTF platform.

C. PicoCTF

PicoCTF[10] is an annual CTF competition run by Carnegie Mellon University, however the organisers released the platform that the competition runs on as open-source, meaning that other organisers are free to use the platform to create and run their own CTF competitions using the platform. It is primarily written using python with JavaScript. Configuration of the platform is done mainly through editing the HTML templates that are provided as part of the platform’s release. It is an incredibly extensive platform that features integrated API’s and integration with a “shell server” that needs to be deployed alongside the CTF platform itself, as this is the server that users will go to and interact with either through the web client, or through a VPN connection in order to solve some of the exploitation challenges. It is a very heavy but incredibly fully featured platform that allows extensive configuration. For deployment, PicoCTF supports manual deployment but also provides vagrant container deployment, allowing bringing up of a platform, shell and web instance through one image- this is the recommended deployment and is the deployment most discussed in the documentation. PicoCTF’s main advantage can also be it’s weakness however, from a customisability perspective, the large and multi-faceted nature of the platform can be overwhelming and make finer customisation complex.

D. NightShade

The NightShade platform[11] is a relatively new CTF platform, that provides a simple and easy to customise challenge platform. It’s based on Django and utilises a mixture of JavaScript for the front-end and primarily Python for handling back-end hosting. In comparison to the other
platforms, challenges for NightShade can be configured directly on the website, once it has been deployed, through the configuration of “super users”, filling the role of site administrators. This makes it an excellent choice for those who want a simple, good looking system that can primarily be configured without having to interact with large codebases. The recommended deployment uses Nginx and Gunicorn as the web server and backend. NightShade’s major advantages are its usage simplicity and cohesive, thematic UI that provides a user, team and scoreboard system without further configuration required. It lacks the extensible capabilities provided by PicoCTF’s API’s and web shells, therefore any challenges that involve exploitation of remote machines will have to be managed separately.

E. FBCTF

The team behind Facebook’s capture the flag competition released the source code for hosting an extremely stylised capture the flag competition using the platform that they created for their event. This is known as both FacebookCTF and FBCTF[12]. It allows for extremely intuitive web-based configuration and supports a wide array of configuration options. While being an extremely heavy platform, it contains a lot of features- including finer granular control, suited for larger-scale events, such as dynamic timers and leader boards, as well as easier administration on a per competition and per task basis without having to delve into any code. It does still lack the ability that PicoCTF provides for an integrated web shell that allows for challenges to be hosted and interacted with directly through the platform, however. This means that challenges involving direct exploitation must be handled separately and linked to on a per challenge basis. The user interface does provide a lot of information, with a stereotypical Hollywood-hacker aesthetic that some may find needlessly complex, the task-setting for example is tied to countries in a global-incident-response style that may not fit all competition themes. A great feature that is built in is support for multiple languages natively. In terms of deployment, this platform is written in hack, an extension of PHP and is extremely well suited to multi-server or multi-container deployment supporting native, vagrant and docker configurations.

F. RootTheBox

The RootTheBox platform[13] is a scoring engine and CTF platform, written in Python. It acts as a hybrid between the standard jeopardy style platforms by adding in new features such as the option to use a banking system, where (in-game) money can be used instead of points to unlock new levels, buy hints to flags, download a target’s source code, or even "SWAT" other players. Password hashes for player bank accounts can also be publicly displayed, allowing competitors to crack them and steal each other’s money. It provides closer to a gamified learning platform than a strictly CTF-oriented platform, combining a videogame-esque environment with realistic challenges that convey knowledge applicable to the real-world, such as penetration testing, incident response, digital forensics and threat hunting. Notably, as well as providing the platform, RootTheBox also provides a fully GUI way of adding “boxes”, i.e. hackable virtual machines into the challenges, making it easy to provide a storied and cohesive pathway for the challenges to follow while providing good administrative access to the platform. In terms of deployment, there are pre-built docker containers that can be deployed easily, however for such a heavy platform it does not provide
any documentation or support for multi-server implementations natively, therefore hosting large events on this platform may cause problems.

VI. Attack-Defence Style Platforms

A. CTF01d

The CTF01d [14] platform is a Python framework that implements a detail, scoring and leader board system for Attack-Defence style capture the flag events. It provides a simple, flag-based submission system where the administrators are able to set a time-based cutoff for flag submission as well as customise the value of the flags to match the deployed machines. One team’s attack flags, are the other team’s defence flags. At the end of the competition, if the other team has not submitted an attack flag, then the other team can get the value of the defence flag added to their score. The same is true in reverse for the attack flags. The platform is written primarily in c++, and the platform supports English and Russian languages. The user interface is clean and simple, with the competition and challenge details being easily available and an intuitive, colour-coded scoreboard. The platform allows for deployment using a simple docker container, making deployment easy. However the large codebase and lack of extensive documentation can make more granular control over the system difficult.

B. ForcAD

ForcAD [15] is a large, extremely extensible attack-defence style CTF platform, that features a large array of functionality, including systems not usually seen in CTF platforms. Along with the standard flag submission, user and team based scoreboards and colour-coded user interfaces, ForcAD utilises a unique to CTF Elo-based ranking system. This uses a system similar to the Elo rank system from chess, which introduces an inflation/deflation handicap system so that teams that are rated higher earn less points per flag if their opponent is significantly lower rated. This ratio is calculated based on how much of a rank difference there is between the two teams. Furthermore, the platform is not built as one monolithic component, rather it is comprised of several smaller modules, including API integration, this allows excellent granular configuration on everything from the scoring system to the checking of flags and load balancing. In terms of deployment, a pre-made docker container can be easily deployed, however each of these services can be installed individually and run from a server directly.

C. iCTF-Framework

The iCTF Framework [16] is an attack-defence cyber security challenge framework created by University College of Santa Barbara in order to facilitate the iCTF competition, the longest-running live educational hacking competition. The developers of the competition released the source code for the platform as open source to allow others to utilise this framework for their own competitions. The framework is formulated of four main components. The database tracks the state of the game, it runs in it’s own VM and communicates with the other components through a REST API, this also allows for extensible integration with user customisable API endpoints. The Gamebot runs the competition, divided into “ticks”. At the beginning of each tick, the Gamebot decides which scripts need to be executed by the scriptbot (e.g., scripts to set flags, retrieve flags, or test services) and writes the schedule in the central database. Then, it extracts from the database the data about the previous tick (e.g., flag submitted and the status of service checks) and computes the points to be assigned to each team. Finally, there is an OpenVPN Router that routes traffic in between teams in the competition. The platform is designed to be deployed on AWS, allowing for
easy deployment and good resource usage scaling depending on the application and load on the platform.

VII. COMMERCIAL PLATFORMS

A. HackTheBox

Fig. 10. The HackTheBox Machine Selection Page

HackTheBox [17] has become a major player in the world of cyber security training. With a focus on hands-on red and blue team security activities, challenges are divided into machines, each with a specific difficulty rating. Similar to other Capture the Flag activities, these have a user level privilege and a root level privilege flag. The points obtained go towards the user’s score, and their team if they belong to one. This is compared to global and national scoreboards. There is a constant rotation of active boxes, that can be accessed for free, however these boxes are shared with other users who can all connect to the same instance, which can result in the public boxes breaking. Interestingly, the machines are tagged with specific labels, with a distribution graph, allowing the user to see the learning topics and challenge focus on a per activity basis. However, HackTheBox places emphasis on the site-provided challenges. There is no way to set custom challenges for exclusive sets of users, and while an academy service is available for learning theoretical and practical topics exists along with labs to provide larger scale challenges- there is little features for creating and sharing individual challenges or running individual events.

B. TryHackMe

Fig. 11. The TryHackMe Hacktivities Selection Page

TryHackMe [18] has also emerged as a platform that provides many benefits of self-organised and self-hosted platforms, with the benefits of first party hosting and community made challenges which are released weekly. Completion of challenge questions rewards the user with points, which go towards the user’s score on a global and national leader board. Along with the pre-made challenges, any user can create and upload a virtual machine, and then set their own questions in a "room", which can then be shared via a code with others, who can simply enter the code and complete the challenges. This provides great options for user-run competitions or events, however it lacks some features that a truly self hosted platform provides- such as API integration with other services for external scoreboards or team validation.

C. Hack.Me

Fig. 12. The Hack.Me Deployable Machine Selection Menu

Hack.Me [19] is a community-driven sandbox platform, it allows users to submit vulnerable virtual machines that can then be deployed by other users and exploited. This allows for easy upload and deployment of challenges that can then be completed. This provides an invaluable service in that it hosts, and provides deployable instances of user-made vulnerable machines. While this is a system that is viable for small scale sharing of exploitable machines, it does lack a lot of the functionality found in competing capture the flag platforms- such as team systems, filterable scoreboards and even organisation of tasks. It is a very open-ended platform and that does have benefits, however- but it's lack of key features suggest it's use-cases are better suited for community or individual usage, as opposed to events or task-based training.
### TABLE 1
A table to compare the features of gamified cyber security training platforms.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Logically Grouped Tasks</th>
<th>Integrated Interactive Environment</th>
<th>Hosting</th>
<th>Individual Deployable Machines</th>
<th>Dynamic Scoring System</th>
<th>Filterable Scoreboard</th>
<th>Extensibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenCTF</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>TinyCTF</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>PicoCTF</td>
<td>Yes</td>
<td>Yes</td>
<td>Self</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>NightShade</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FBCTF</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RootTheBox</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CTF01d</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ForcAD</td>
<td>Yes</td>
<td>Yes</td>
<td>Self</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>iCTF</td>
<td>Yes</td>
<td>No</td>
<td>Self</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>HackTheBox</td>
<td>Yes</td>
<td>Yes</td>
<td>Commercial</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TryHackMe</td>
<td>Yes</td>
<td>Yes</td>
<td>Commercial</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Hack.Me</td>
<td>Yes</td>
<td>Yes</td>
<td>Commercial</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Portswigger</td>
<td>No</td>
<td>No</td>
<td>Commercial</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>RootMe</td>
<td>Yes</td>
<td>No</td>
<td>Commercial</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**D. Web Security Academy**

Portswigger, the company behind the BurpSuite security tool, offer a free online platform for interactive, practical cyber-security training called the web security academy [20]. It provides a clean interface with deployable instances of machines that are accessible to the user without needing to utilise a VPN. The tasks are broken down into individual "labs" each focusing around a specific area of web security, with interactive tutorials associated with each activity and unassisted challenges at the end of each "lab" to consolidate the user’s learning.

![Fig. 13. The Web Security Academy Splash Page](image)

Completing these sections gives the user points, which are added to their overall score, displayed on a leader board along with the number of completed rooms, allowing the user to accurately track their progress. The downsides of this platform are that it does not include a team or group function and features no way for any user to act as a "leader" and assign tasks, meaning only first party resources are able to be used. Furthermore, the focus is clear even from the name. Only web security, exploitation and surrounding theory is covered on this platform, there is no material on other topic such as binary exploitation or digital forensics available.

**E. RootMe**

![Fig. 14. The RootMe Challenge Selection Page](image)

The RootMe [21] platform offers a wide variety of challenges for the users to complete, grouping tasks into individual "tasks" that contain the task material. Similarly to other commercial offerings, there is a range of task material from applications to reverse engineer or exploit, to simple websites for web security challenges to entire deployable virtual networks. Completion of challenges rewards the user with points that increase their score on the global leader board. RootMe also offers an educator mode, which allows an organiser or class leader to set tasks, or create events for specific users— including time limits and cutoffs for submission.

**VIII. FINDINGS**

**A. Table of Comparison**

If we draw a table of comparison, comparing our essential feature sets outlined above, to the platforms that we have
reviewed, we are able to see which of the platforms contains the most relevant features and is therefore most likely to meet the needs of the most users [Table 1]. This shows the merit of applying our proposed feature set as the basis for comparing the applicability of each of these platforms.

IX. CONCLUSION

Ultimately, from our survey of the currently available gamified cybersecurity training platforms, we have found that there are three six main feature sets that are shared by most platforms. These are, that tasks should be logically grouped and have interactive task material easily available to the user. That tasks, users and teams should be easily administrable from the platform. Time-based submission cutoffs should be simple to set, and adjust by the administrators. The scoring system should be simple to understand and display, with the reward for harder tasks being higher than easier ones. Furthermore, a dynamic scoring system based on team rating should be supported for head to head Attack/Defence scenarios. Furthermore, creation and deployment of interactive, exploitable machines should be focused around containerised solutions or virtual machines. Utilising this criteria, we can show from our table of comparison that for a self-hosted solution, PicCTF is the best platform to use, as it meets the largest number of these criteria and for the commercial offerings, TryHackMe, HackTheBox or RootMe all offer similar solutions, and can therefore be used depending on which features are desired for the organiser as they all a large number of, but slightly differing feature criteria. We also identified an important area of expansion, which is the extensibility of each of these platforms, such as API integration for specific user or team actions. We believe this is something that a next-generation platform should provide.

ACKNOWLEDGMENT

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no. 833673. This work reflects authors’ view and Agency is not responsible for any use that may be made of the information it contains.

REFERENCES