VulnerSec: A Flexible, Automated and Open-Source Cybersecurity Framework

Evgenija Krajchevska
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius University
Skopje, North Macedonia
evgenija.krajchevska@finki.ukim.mk

Vojdan Kjorveziroski
Faculty of Computer Science and Engineering
Ss. Cyril and Methodius University
Skopje, North Macedonia
vojdan.kjorveziroski@finki.ukim.mk

***Abstract*—Hands-on cybersecurity training is crucial for developing practical skills, yet existing platforms often present limitations such as being closed-source, requiring complex setup or relying on outdated vulnerabilities that become unusable over time. To tackle these issues, we introduce an open-source framework designed to create flexible, customizable and reproducible cybersecurity scenarios. It provides a modular approach, allowing deployment of vulnerable environments with minimal setup overhead, and addresses legacy software vulnerabilities making them accessible even on modern systems. The decoupling of the provisioning and deployment processes ensures greater adaptability, allowing the same configurations to be applied across diverse infrastructures with minimal adjustments while widely adopted tools are used to simplify these processes. The framework was validated in a university cybersecurity course where students participated in vulnerability discovery and exploitation exercises. Future work will primarily focus on expanding the vulnerability set, implementing profile-based configurations that provide customizable scenarios based on complexity levels as well as implementation of network monitoring and generation of datasets for the improvement of intrusion detection and threat mitigation processes.**

***Index Terms*—hands-on cybersecurity, software vulnerabilities, framework, automation, virtualization**

I. Introduction

With the rising demand for IT professionals proficient in ever-changing technologies and tools, it is evident that practical work in computer science has become essential, even in introductory courses accommodating learners of diverse backgrounds and varying skill levels [1]. Hands-on exercises and project-based learning constitute an important aspect of both formal and informal education, as they immerse students in real-world tasks and foster active engagement with course material. Extensive studies show that interactive, hands-on activities boost engagement and performance in science and cybersecurity courses alike [2] [3].

When it comes to cybersecurity, research highlights that structured cybersecurity exercises improve both technical proficiency and decision making skills by immersing learners in realistic threat scenarios [4]. Challenge-based learning, especially in the form of Capture The Flag (CTF) exercises, can also be seen as an effective methodology in cybersecurity education, demonstrating improvements in practical skills, motivation, and participation of students when faced with real-world scenarios [5].

The main obstacle when organizing hands-on exercises is that their planning and infrastructure setup takes too much time from the educators. This is especially the case with cybersecurity since the integrity of the entire system must be considered. Unlike general computing tasks, these exercises require secure isolated environments to avoid compromising institutional networks and demand constant maintenance and updates to keep pace with software releases. Additionally, each student requires their own personalized environment, further complicating things. Providing every learner with an isolated virtual machine instantiates what Cantelli-Forti and Colajanni describe as the stateful paradigm, where the environment preserves interaction state across sessions, thereby intensifying both setup and maintenance overhead [6]. Some of these challenges can be alleviated by using proprietary services and platforms, albeit they introduce their own drawbacks.

One common limitation of commercial solutions is that subscription fees can be prohibitively expensive for institutions with limited budgets, particularly in developing countries, while the lack of source code access makes it difficult for instructors or researchers to fully understand or adapt the underlying security controls and architectures. Furthermore, most proprietary offerings provide only a fixed set of scenarios or challenges, preventing educators from personalizing content to specific learning objectives or incorporating novel vulnerabilities. By contrast, open-source solutions are freely available to both instructors and researchers, stimulating swift iteration and collaborative development through community-driven repositories. Such solutions often embrace modular design and integrate seamlessly with a variety of hypervisors, operating systems, or cloud platforms, making it easier to keep pace with evolving requirements.

Aside from being a valuable asset in educational practices, these frameworks play a critical role in cybersecurity research, specifically in studying real-world attack behaviors and defense strategies. Due to their isolated nature, vulnerable environments are often used to observe attack vectors, analyze vulnerabilities, and simulate realistic attack-defense scenarios. Multiple recent efforts highlight the importance of novel research in this domain. Simulation frameworks remain an

effective way to model multi-stage attacks and observe the interplay between complex network vulnerabilities, security devices, and traffic patterns [7]. Modeling frameworks take into account the whole lifecycle of a cyber defense, starting from proactive prevention and detection all the way to thorough incident analysis and recovery which allows researchers to evaluate how various defensive measures perform under realistic threat conditions [8]. By simulating various threat vectors in controlled yet scalable testbeds, cybersecurity frameworks enable researchers to evaluate different defense mechanisms without putting production environments at risk.

The goal of this paper is to present an open-source framework for generating and customizing virtual environments to support hands-on cybersecurity exercises. By simplifying infrastructure and tooling, instructors can deploy vulnerable software with greater flexibility, relying on standardized tools with few prerequisites. Additionally, a container-based approach is adopted to overcome scenario rot and ensure that legacy vulnerabilities remain usable for teaching and research, regardless of host system updates. By offering this solution under a permissive open-source license, we aim to encourage collaboration, adaptability and sustainable use for educators and researchers alike.

The remainder of the paper is structured as follows. Section II discusses existing platforms and scenarios for hands-on exercises, highlighting related work in the field. Section III presents our proposed requirements for a flexible framework, focusing on core aspects such as: customizability, openness, sustainability and ease of use. In Section IV, we introduce the technical architecture and design choices of the presented solution, illustrating how it meets the requirements defined earlier. Section V describes the practical validation of our approach in an introductory cybersecurity course, outlining the deployment process and initial results. Finally, Section VI concludes the paper by summarizing key findings and suggesting directions for future work.

II. Related Work

Various platforms have emerged for integrating vulnerable virtual machines into cybersecurity education, each providing an isolated environment in which students can hone their practical skills. These solutions, frequently adopted in university courses or Capture The Flag (CTF) competitions, enable learners to engage directly with security threats in a controlled setting.

Recent research has explored diverse strategies for the deployment of cybersecurity labs. Russo et al. introduce a gamified cybersecurity education program structured around progressive engagement levels [9]. The entire infrastructure is self-hosted within the faculty's network, ensuring full control over the environment and security configuration. It simulates enterprise-like networks, including firewalls, DNS servers, and client systems, where students individually defend against scripted attacks. While this approach provides high realism and full control over the environment, it introduces remarkable challenges related to manual setup, scalability limitations and resource-intensive maintenance.

In contrast, the CyberArena framework focuses on a scalable, cost-effective architecture utilizing the cloud [10]. Students remotely access isolated lab environments through a browser-based interface while their assessments are carried out through automated scripts. Instructors can add new challenges with the required software, data, user configurations, and automated scripts needed for lab interaction. Proper setup including all necessary dependencies and security settings can be time-intensive. More recently, Nelson et al. introduced DOJO [11], an open-source platform for applied cybersecurity education which incorporates both self-hosted and centrally-hosted deployment models, easing the infrastructure burden for educators. It addresses numerous aspects of lab creation, such as improved accessibility by allowing students to interact with challenges directly through browser-integrated tools as well as providing scalable and reproducible challenge environments by leveraging Docker containers while isolating students' activities. However, it faces constraints in customizing more complex scenarios, particularly those involving legacy vulnerabilities or kernel-level configurations, which often require additional manual setup.

We next examine a selection of well-known platforms that illustrate current practices in the area. One such platform is SecGen [12], an open-source software to automatically provision virtual machines with custom vulnerabilities. Its key-strength lies in the ability to define reproducible scenario-based challenges using a declarative approach. Another open-source project is Vulhub [13], which leverages a container-based approach in the deployment of vulnerabilities, allowing minimal overhead and rapid deployment. It is commonly included as part of cyber ranges developed within this area [14]. Vulnhub [15], by contrast, is not a framework but a community-driven repository hosting a variety of pre-built vulnerable virtual machines that mimic real world security flaws. These machines are frequently used in educational settings to develop penetration testing skills through simulated scenarios. Finally, some platforms deliver "Cybersecurity Labs as a Service", such as CLaaS [16], which provides remote access to pre-configured cybersecurity labs through a web interface, eliminating the burden of local infrastructure management.

Despite the utility of these platforms, each exhibits certain drawbacks that can limit their efficiency in academic contexts. Some rely on older frameworks and require complex scripting, others offer only fixed scenarios without providing customizable environments and a few like CyberArena or CLaaS demand cloud-based infrastructures, potentially limiting accessibility. Moreover, very few solutions address both modern containerized vulnerabilities and legacy exploits that may fail on recent operating systems. Table I outlines the main features and limitations of these solutions, including their deployment models, open-source accessibility, level of customization and hosting requirements.

TABLE I
FEATURE COMPARISON OF CYBERSECURITY TRAINING PLATFORMS

Feature	SecGen	Vulhub	Vulnhub	CLaaS
Type	VM-based scenarios	Container-based scenarios	VM Repository	Cloud-based VM scenarios
Scenario Setup	Puppet	Docker-compose	None (Prebuilt)	VMware/ESXi
Runtime Environment	Virtual machines	Containers	Virtual machines	Virtual machines
Customizable Scenarios	Yes	Yes	No	No
Open-source	Yes	Yes	No	No
Self-hosted	Yes	Yes	Yes	No

III. REQUIREMENTS FOR A FLEXIBLE FRAMEWORK FOR HANDS-ON SCENARIOS

To design an effective framework for hands-on cybersecurity, it is crucial to establish a set of core principles that guarantee adaptability and ease of use. Many existing solutions, as discussed previously, address certain aspects of cybersecurity but often lack the flexibility needed to become truly sustainable.

A. Functional Requirement 1: Scenario Customization

The most significant functional requirement is scenario customization. A well-structured cybersecurity framework must allow users to define configurations dynamically. Generating the challenges using such an approach assures that exercises remain engaging and educationally valuable. Additionally, a cybersecurity framework should be scalable and adaptable, allowing instructors and researchers alike to modify and extend scenarios to accommodate evolving objectives. Hands-on environments must also be resilient to obsolescence. Without proper design considerations, lab scenarios risk becoming outdated, limiting their educational effectiveness. Properly configured environments should not take too much time off of users' hands, as they should minimize complexity and overhead in a way that supports managing and deploying scenarios with ease.

B. Functional Requirement 2: Open-Source Access

An equally important functional requirement is for the platform's source code to be publicly available under a permissive open-source license. A wide range of cybersecurity training solutions have already been developed, yet a common limitation among them is their closed-source nature, restricting the ability to modify or extend their features beyond their original design. This is particularly evident in frameworks developed exclusively for specific institutions, adapted to meet the internal requirements of a single faculty or training program. In addition, an increasing number of commercial cyber ranges operate on a proprietary, subscription-based model, offering training as an online service to individuals seeking to strengthen their skills [17]. By comparison, a solid cybersecurity lab framework should be open-source, inspiring educators and researchers to adjust scenarios, collaborate across institutions and shape scenarios to meet their requirements. Open-source solutions promote long-term sustainability, as they enable a shared development model where improvements and new challenges can be contributed by a broader academic and professional community. By encouraging collaboration among instructors, such frameworks not only enhance the quality and diversity of cybersecurity scenarios but also ensure that training environments remain up to date and widely accessible over time.

C. Functional Requirement 3: Long-Term Sustainability

Cyber-security labs often suffer scenario rot, where exploits break because of software updates, missing dependencies, or vanished resources. We define the third functional requirement as the ability to handle different vulnerability types, including legacy ones. For a framework to be long-term sustainable, it must guarantee that both the scenario and its execution environment remain reproducible over time. Support for multiple diverse runtime environments, each providing isolation features, can prevent compatibility issues and maintain accessibility to older vulnerabilities. A one-size-fits-all approach is insufficient when dealing with a broad range of vulnerabilities, often requiring a flexible deployment strategy, allowing even complicated vulnerability scenarios involving multiple components to be deployed. This approach mitigates dependency issues and preserves older vulnerabilities for continued use. By incorporating support for multiple runtime environments, such a framework offers educators and researchers the flexibility to customize scenarios to suit a variety of necessities.

D. Functional Requirement 4: Ease of Use

Lastly, a functional requirement that should not be overlooked is ease of use. Ideally, frameworks should be straightforward to configure without delving too much into technical details, enabling even educators or researchers who may lack technical skills to navigate the platform with ease. At the same time, the framework must guarantee strict inter-user isolation and protect the host infrastructure by providing practical security safeguards, so students cannot break out of their assigned virtual machines or containers while instructors retain a seamless workflow. The setup process should be intuitive and require minimal manual intervention, relying on automated workflows and widely adopted tools. By simplifying the deployment process the framework remains accessible across diverse educational environments, reducing the technical burden and permitting instructors to focus on various aspects of cybersecurity training and research.

The following section presents a framework that embodies these principles, delivering a scalable and adaptable solution for hands-on cyber security training.

TABLE II
OVERVIEW OF EXAMPLE VULNERABILITIES IN VULNERSEC

Vulnerability Name	Decription	Vulnerability Type	Runtime Environment
Crackable User Account	Weak user credentials	Weak password	Virtual machine
SSH Root Login	SSH root access enabled	Misconfiguration	Virtual machine
NFS Root Share	NFS root export	Misconfiguration	Virtual machine
Netcat Backdoor Chroot Escape	Netcat in chroot environment	Misconfiguration, Backdoor	Virtual machine
UnrealIRC 3.2	Vulnerable IRC Server	Backdoor	Container
ProFTP 1.3.3c	Vulnerable FTP Server	Backdoor	Virtual machine
VsFTP 2.3.4	Vulnerable FTP Server	Backdoor	Virtual machine
Wordpress 3.x	Outdated, vulnerable CMS	Web Application Exploit	Virtual machine
Grafana 8.0.0	Vulnerable Grafana Version	Path Traversal	Container

IV. THE VULNERSEC PLATFORM FOR HANDS-ON SCENARIOS

For a platform to be flexible, it must be capable of adapting easily to various teaching requirements and technical constraints. In alignment with the importance of scenario customization, our proposed framework, VulnerSec, addresses the need for enhanced adaptability by supporting dynamic, easily customizable cybersecurity scenarios without substantial overhead. Instead of relying on complex, agent-based frameworks that require dedicated infrastructure, we employ Ansible which offers simplicity, agentless operation, human-readable syntax and modular design paving the way for customization and smooth development. VulnerSec supports both virtual machines and containerized environments, maximizing compatibility across diverse infrastructures. Virtual machines are essential for comprehensive system simulation, permitting in-depth interaction with operating system components, while containers offer an efficient method for swift deployment of vulnerable services. This approach fulfills the first functional requirement for scenario customization (FR1).

Consistent with our commitment to encouraging cooperation and transparency, VulnerSec is released under a permissive open-source license, inspiring educators, researchers and practitioners alike to reuse, contribute to and expand the platform. The complete implementation as well as all scenarios and roles is publicly available on Github [18], thus satisfying the second functional requirement for open-source access (FR2). Table II provides an overview of the example vulnerabilities implemented to date, including a concise description and the associated vulnerability type.

Aware of the importance of long-term usability, we specifically target the handling of scenario rot, which happens when outdated vulnerabilities become unusable due to incompatible software versions or broken dependencies. To counteract this, the framework incorporates containerized deployments, which encapsulate legacy software in isolated, reproducible runtime environments, thus satisfying the third functional requirement for handling long-term sustainability (FR3). For this purpose, we use a role that sets up Docker on the target machine. Vulnerable containers can subsequently be launched using a predefined container template with customizable configuration parameters. Nevertheless, software tools deployed today are readily available as images, so this versatile approach further eases the setup, consisting only of specifying an image and the configuration for the vulnerable container. It allows VulnerSec to be compatible and incorporate third-party containers which are already available, such as the container repository published by VulHub. This further demonstrates the flexibility of the proposed framework.

In accordance with the fourth functional requirement for ease of use (FR4), each vulnerability is implemented as an Ansible role, structured according to best practices. These environments are self-contained where each role envelopes a specific vulnerability, ensuring educators and researchers can deploy complex scenarios without navigating tangled dependencies. Deploying a vulnerability is as simple as running the corresponding Ansible role in the virtualized environment. One notable advantage of the VulnerSec framework is that vulnerabilities can be customized to be largely platform-agnostic, leveraging Ansible's modularity and abstraction capabilities. Because each exploit is encapsulated as an independent Ansible role, instructors can swap modules, override variables, or publish new scenarios without touching the surrounding playbooks. [20]. The declarative format permits instructors to easily adapt scenarios to different base operating systems or infrastructures without requiring significant rewrites or modifications. Recent research suggests that Ansible is highly regarded for its ease of use, attributed to its agentless design and straightforward YAML playbooks. Users with minimal prior experience in configuration management tools have reported smoother onboarding and faster initial deployment processes [19]. Its usability is further reflected in its continued adoption within the cybersecurity domain. Chouliaras et al. conducted a systematic survey of ten cyber ranges that were developed in the last decade, analyzing their characteristics based on deployment strategies, configuration management approaches, and automation tools. Their findings indicate that Ansible is among the most widely adopted solutions for automating the deployment and configuration of cybersecurity scenarios, highlighting its prevalence within the field. Using this approach enables instructors and researchers alike to quickly spin up reproducible and isolated virtual machines from a single configuration file, significantly reducing deployment time.

Beyond accelerating the generation and maintenance of vulnerable instances, the platform's design also lays the groundwork for network monitoring and data collection. With the integration of standard monitoring tools, ranging from packet

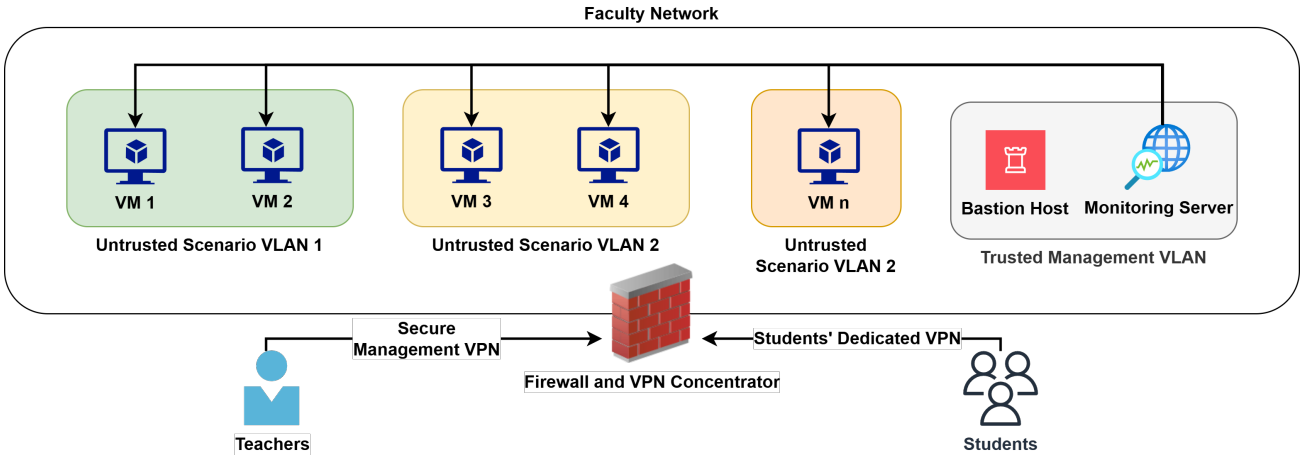

Fig. 1. Visual representation of the network architecture used during the validation stage

sniffers to log analyzers, platform administrators could capture detailed traces of all traffic generated within these isolated environments. This data can then be annotated to identify attempted attacks, the specific vulnerabilities exploited, and whether each intrusion attempt succeeded or failed. Such annotations would offer invaluable insight not only for validating new detection algorithms, but also for building specialized datasets that reflect realistic attack patterns. As a result, the researchers would be able to examine the effectiveness of different defensive techniques under numerous threat scenarios. These integrated monitoring capabilities thus would extend VulnerSec's utility, offering a customizable testbed where both educators and practitioners can structure, assess, and advance security tools and practices in a contained environment.

V. Practical Validation

The proposed VulnerSec framework was practically validated during an introductory course at the Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje, involving 26 students who participated in the hands-on exercises within a one-week timeframe. We initiated the validation process by instantiating 13 virtual machines, each configured with one to three vulnerabilities. The participants' primary objective was to discover the machines and exploit the vulnerabilities. To accommodate remote participation, students conducted the exercises from their homes, connecting securely to the infrastructure via dedicated virtual private network (VPN) profiles. This ensured controlled and isolated interactions with the vulnerable environments while maintaining the integrity and security of the host network.

The infrastructure was hosted on—premise and comprised an OPNSense firewall [21], a VPN server for secure remote student access, an ESXi hypervisor, and Docker container runtimes instantiated in the virtual machines themselves for hosting certain vulnerabilities. All of the virtual machines were continuously monitored to ensure their availability and to verify that the students adhered to the rules and did not make changes to the configuration of the deployed services. While local testing was conducted using VirtualBox, the actual deployment within the faculty's infrastructure utilized the Vagrant provider for VMware. Once instantiated, a bastion host was used to execute the customization playbooks, thus acting as a unified place from where all involved educators could manage the environment.

It should be noted that the provisioning process is decoupled from the deployment process, allowing the same Ansible-based configuration to be applied regardless of the virtualization provider. Fig. 1 provides a visual overview of the described infrastructure setup, illustrating the network topology and undertaken isolation measures.

All of the vulnerabilities were deployed by executing an Ansible playbooks, comprised of dedicated roles corresponding to the scenario at hand. This role-based approach enables flexibility, allowing us to configure different scenarios dynamically. Each virtual machine combined a variety of vulnerabilities, including misconfigured services, outdated web applications and containerized legacy software backdoors. Moreover, a dedicated flag-generation mechanism was implemented, enabling students to verify successful exploitation in a structured manner. Text-based flags are placed in predetermined locations of the file-system, which become accessible to the attacker once the corresponding vulnerability has been successfully exploited. This acts as a further proof that the student has indeed successfully exploited the vulnerability and thus completed the challenge.

To ensure strict isolation from the rest of the network infrastructure, the virtual machines hosting the vulnerabilities were placed in dedicated virtual local area networks (VLANs). The firewall restricted their network access, blocking all inbound and outbound connectivity, with the only exception being the virtual private network subnet. This allowed the students to probe the virtual machines for vulnerabilities, as well as establish reverse connections to their own devices as part of the exploitation process. Multiple VLANs were utilized to

simulate a more elaborate network environment, reminiscent of the real-world.

VI. Conclusion

Cybersecurity education and research requires hands-on practice with realistic attack scenarios, yet existing solutions are often highly complex, non-customized, closed-source, or suffer from scenario degradation. To address these challenges, this paper introduced VulnerSec, an open-source framework for deploying customizable and reproducible virtualized and containerized vulnerable environments. Our contributions include a modular approach that allows flexible scenario creation, a solution to scenario rot by leveraging containerized deployments, and an easily deployable infrastructure, decoupling the provisioning and deployment processes. This framework enables users to efficiently manage cybersecurity environments without extensive overhead.

Future work will primarily focus on expanding the range of vulnerabilities by introducing new exploits as well as developing predefined vulnerability profiles which will allow instructors to generate randomized attack scenarios with different levels of difficulty. Moreover, future possibilities include the generation of events datasets where each captured event is traceable to a specific vulnerability, exploit type, or traffic pattern. Training data of this nature will further contribute to the development of Intrusion Detection Systems (IDS) and other defensive algorithms since a variety of machine learning approaches such as supervised, unsupervised or semi-supervised learning benefit from labeled, authentic scenarios. The flexibility of the framework allows researchers to iterate rapidly, adding or modifying test scenarios to reflect emerging threats and new defense mechanisms. At the same time, it presents the potential for crowdsourcing additional attack traces or analytics drawn from their own testing scenarios. This collective approach would enrich the data beyond what a single group could create on its own, thereby improving the overall quality and research value of the dataset.

As the initial implementation was tested on a limited group of students, future work will also focus on large-scale deployments to evaluate scalability and performance in distributed environments. By releasing VulnerSec under a permissive open-source license, we invite collaboration from the cybersecurity community to refine and extend the platform for the evolving needs of cybersecurity education.

VII. Acknowledgement

This work was partially financed by the Faculty of Computer Science and Engineering at the Ss. Cyril and Methodius University in Skopje, under the CYBER-EAGLE project.

This research was in part sponsored by the NATO Science for Peace and Security Programme under grant id. G7593

References

[1] CompTIA, "State of the Tech Workforce 2024," CompTIA, Downers Grove, IL, USA, Mar. 2024. [Online]. Available: https://www.comptia.org/content/research/state-of-the-tech-workforce

[2] C. O. Ekwueme, E. E. Ekon, and D. C. Ezenwa-Nebife, "The Impact of Hands-On-Approach on Student Academic Performance in Basic Science and Mathematics," Higher Education Studies, vol. 5, no. 6, pp. 47-51, 2015. doi:10.5539/hes.v5n6p47.

[3] L. Williams, E. Anthi, Y. Cherdantseva, and A. Javed, "Leveraging Gamification and Game-Based Learning in Cybersecurity Education: Engaging and Inspiring Non-Cyber Students," J. Colloq. Inf. Syst. Secur. Educ., vol. 11, no. 1, Winter 2024.

[4] M. Karjalainen and T. Kokkonen, "Review of Pedagogical Principles of Cyber Security Exercises," Advances in Science, Technology and Engineering Systems Journal, vol. 5, no. 5, pp. 592–600, 2020, doi: 10.25046/aj050572.

[5] S. V. Cole, "Impact of Capture The Flag (CTF)-style vs. Traditional Exercises in an Introductory Computer Security Class," Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2022), Dublin, Ireland, July 2022, pp. 470–476, doi: 10.1145/3502718.3524806.

[6] [22] A. Cantelli-Forti and M. Colajanni, "Adversarial Fingerprinting of Cyber Attacks Based on Stateful Honeypots," Proc. Int. Conf. Computational Science and Computational Intelligence (CSCI), pp. 19–24, Dec. 2018, doi: 10.1109/CSCI46756.2018.00012.

[7] S. Kara, S. Hizal, and A. Zengin, "Design and implementation of a DEVS-based cyber-attack simulator for cyber security," Int. J. Simul. Model., vol. 21, no. 1, pp. 53–64, 2022, doi: 10.2507/IJSIMM21-1-587.

[8] D. Kim, M. K. Ahn, S. Lee, D. Lee, M. Park, and D. Shin, "Improved cyber defense modeling framework for modeling and simulating the lifecycle of cyber defense activities," IEEE Access, vol. 11, pp. 114187–114206, Oct. 2023, doi: 10.1109/ACCESS.2023.3324901.

[9] E. Russo, M. Ribaudo, A. Orlich, G. Longo, and A. Armando, "Cyber Range and Cyber Defense Exercises: Gamification Meets University Students," Proc. 2nd Int. Workshop on Gamification in Software Development, Verification, and Validation (Gamify '23), San Francisco, CA, USA, Dec. 2023, pp. 1-9, doi: 10.1145/3617553.3617888.

[10] C. Tunc and S. Hariri, "CyberArena: An Open-Source Solution for Scalable Cybersecurity Labs in the Cloud," Journal of Internet Services and Information Security (JISIS), vol. 5, no. 4, pp. 41-59, Nov. 2015.

[11] C. Nelson and Y. Shoshitaishvili, "DOJO: Applied Cybersecurity Education In The Browser," Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024), Portland, OR, USA, March 2024, pp. 930-936. DOI: 10.1145/3626252.3630836.

[12] Z. C. Schreuders, T. Shaw, M. Shan-A-Khuda, G. Ravichandran, and J.Keighley, "Security Scenario Generator (SecGen): A framework for generating randomly vulnerable rich-scenario VMs for learning computer security and hosting CTF events," 1st UK Workshop on Cybersecurity Training & Education (VIBRANT 2015), Liverpool, UK, 2015.

[13] Vulhub, "Vulhub - Pre-Built Vulnerable Docker Environments," [Online]. Available: https://vulhub.org/. [Accessed: 11-Mar-2025].

[14] A. Mills, J. White, and P. Legg, "GoibhniUWE: A Lightweight and Modular Container-Based Cyber Range," Journal of Cybersecurity and Privacy, vol. 4, pp. 615–628, 2024, doi: 10.3390/jcp4030029.

[15] VulnHub, "VulnHub: Providing Vulnerable Virtual Machines for Learning and Practicing Cybersecurity," [Online]. Available: [https://www.vulnhub.com]. [Accessed: 10-Mar-2025].

[16] C. Tunc and S. Hariri, "CLaaS: Cybersecurity Lab as a Service," Journal of Internet Services and Information Security (JISIS), vol. 5, no. 4, pp. 41-59, Nov. 2015.

[17] Cloud Range, "The Leading Cyber Range as a Service," Cloud Range. [Online]. Available: https://www.cloudrangecyber.com/. [Accessed: 12-Mar-2025].

[18] E. Krajchevska, VulnerSec: A Flexible, Automated and Open-Source Cybersecurity Framework, GitHub. [Online]. Available: [https://github.com/ekrajchevska/VulnerSec]. [Accessed: 12-Mar-2025].

[19] A. B. Ojel and J. I. Teleron, "Configuration Management and Automation Tools: A Comparative Analysis and Overview," International Journal of Advanced Research in Arts, Science, Engineering and Management (IJARASEM), vol. 12, no. 1, pp. 174-185, Jan.-Feb. 2025.

[20] N. Chouliaras, G. Kittes, I. Kantzavelou, L. Maglaras, G. Pantziou, and M. A. Ferrag, "Cyber Ranges and TestBeds for Education, Training, and Research," Applied Sciences, vol. 11, no. 4, p. 1809, 2021. [Online]. Available: https://doi.org/10.3390/app11041809

[21] OPNsense® a True Open Source Security Platform and More - OPNsense® Is a True Open Source Firewall and More. Retrieved March 14, 2025, from https://opnsense.org/