BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2026
http://excel.fit.vutbr.cz

CTF Framework for Ethical Hacking Education

Hung Do*

Abstract

CTF exercises effectively teach cybersecurity, but infrastructure challenges limit adoption in educational settings. Traditional VM-based deployments consume excessive resources, scale poorly, and require constant maintenance through error-prone shell scripts. We developed a resource-efficient framework that replaces VMs with lightweight containers, reducing memory consumption by up to 94% at idle and up to 83% under full load while simplifying deployment and management. The open-source framework combines *fit-ctf* (orchestration library), *fit-rendezvous* (user platform), and *fit-ctf-virt* (deployable stack) for educators to self-host on bare-metal servers without external dependencies. Key features include per-student unique challenge scenarios generated via templating to prevent flag sharing, flexible assignment of challenge sets to each student, and automatic flag validation to streamline grading. This enables institutions to scale CTF assignments cost-effectively while reducing instructor burden.

*xdohun00@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

Capture the Flag is a popular exercise in cyber-security which combines education and gamification. Users are solving challenges that involve known vulnerability principles, or simulation of real-world environments containing vulnerable or incorrectly configured systems. Users are tasked to exploit these vulnerabilities to get access to a hidden *flag* which is typically a string with strictly defined format.

Our university runs a Capture the Flag assignment in the cyber-security course Information System Security. Students gain access to a sandbox environment and are tasked with exploring, identifying and exploiting vulnerabilities within a system to retrieve flags. In the past several years, the CTF infrastructure heavily relied on virtual machines where each student was assigned a *login node* VM from which they access *target node* VMs containing challenges. Both the infrastructure and CTF challenges are designed and managed by teaching assistants.

This CTF architecture has resulted in several technical difficulties. The main problems involve:

- **inefficient resource utilization** in idle, as login nodes were continuously running and consuming unused CPU and memory resources,
- **hitting resource ceiling** near the deadline as more students actively work on their assignment,
- **difficulty of iterating and maintaining the CTF configuration** as the deployment was all written in shell script,
- **lack of logging and monitoring**,
- **manual submission assessments**, and
- **SSH keys distribution via e-mail**.

2. Existing Solutions

We investigated existing CTF solutions to understand how they address our problems. While suitable for running short-term competitions, they lack features required for educational settings, particularly mechanisms to prevent flag sharing and ensure per-student challenge isolation[1]. Most platforms are optimized for event-based competitions rather than semester-long assignments with diverse learning objectives.

Web platforms like *CTFd*[1] and *picoCTF*[2] provide scoring and team management but do not handle infrastructure, as administrators must manage container deployment separately. Commercial services like *Hack The Box*[3] charge per student and restrict challenges to their catalog. Infrastructure-focused tools such as *kCTF*[4]

[1] https://ctfd.io/
[2] https://picoctf.org/
[3] https://www.hackthebox.com/
[4] https://google.github.io/kctf/

require Kubernetes expertise and resources inappropriate for educational deployments.

Universities have developed custom CTF systems, with some publishing their approaches (e.g., MIT[2], University of Maryland[3]). However, most keep solutions closed-source or unpublished, limiting reusability. We found no open-source solution combining both a user platform and infrastructure toolkit designed for classroom deployment with anti-cheating mechanisms.

3. The FIT CTF Framework

The framework consists of three open-source components: *fit-ctf*, a Python library handling resource management (projects, users, enrollment), container orchestration, and scoring logic; *fit-rendezvous*, a Terminal User Interface (TUI) accessible via SSH for students to manage their challenges and submit flags; and *fit-ctf-virt*, a pre-configured deployable stack including database, monitoring (Prometheus and Grafana), and all dependencies.

The infrastructure uses Podman Compose to manage containers. We chose Podman primarily for its daemonless architecture and rootless container support, which enables simpler deployment and enhanced security. The TUI platform is designed to be lightweight and simple to deploy. It avoids SSL overhead, uses fewer resources than web interfaces, and relies on native libraries rather than external API calls. The deployable stack includes a central database for managing projects and user enrollment, plus integrated monitoring via Prometheus and Grafana to track system performance.

All challenges are organized as modular units called *scenarios*. Each scenario represents a complete environment with multiple containers that must run together. For example, a scenario might include a web server, database, and logging service. If any service is missing, the flag cannot be retrieved. This modular design makes scenarios portable and reusable across students and projects.

To prevent simple cheating such as sharing flags between students, scenarios use templating. When a scenario is deployed to a student, templates inject runtime values, e.g. project names, usernames, file paths, and environment variables which ensure that each student receives a unique flag value even though they solve the same challenge. Administrators can assign different challenge sets from the pool of scenarios to each student. The platform automatically validates submitted flags and provides feedback to teaching staff for assessment.

This approach directly solves the problems described in section 1. Testing on identical hardware (i5-9500T, 16GB RAM) demonstrates that containers consume significantly fewer resources than VMs: at idle, memory usage drops from 9GB to 500MB (up to 94% reduction); under full load (10 active users), memory usage is 2.7GB compared to 16GB+ for VMs (up to 83% reduction). Additionally, containers eliminated the need for swap, while VM deployments required swap at maximum load. Since students manage their own clusters and control when instances run, idle resource consumption is minimized. Scenario-based templating eliminates the maintenance burden of shell-script deployments while enabling flexible, repeatable challenge management.

4. Conclusion

We developed a containerized CTF framework addressing resource efficiency and scalability challenges in educational environments. By leveraging application containers instead of full virtual machines, our solution significantly reduces resource overhead while enabling on-demand cluster management. The modular scenario-based architecture with automated flag validation and per-student unique configurations reduces administrative burden and mitigates cheating risks. Our benchmarking on i5-9500T hardware demonstrates up to 94% memory reduction at idle (9GB to 500MB) and up to 83% reduction under maximum load (16GB+ to 2.7GB), while eliminating the need for swap—a critical bottleneck in VM deployments. The `fit-ctf`[5] framework is released as open-source software, enabling institutions to self-host CTF events on single bare-metal servers without external cloud dependencies. This provides educators with a practical, maintainable, cost-effective solution that scales with student enrollment.

Acknowledgements

I would like to thank my supervisors Ing. Jakub Reš and Ing. Zbyněk Lička for guidance and providing resources for development and testing. Their feedbacks and ideas helped to improve the functionality and usability of the framework.

References

[1] J Vykopal, V Svabensky, and E Chang. Benefits and pitfalls of using Capture the Flag Games in university courses. *arXiv (Cornell University)*, 4 2020.

[5] https://github.com/hungdojan/fit-ctf.git

[2] Eric J. K. Jackson, Matt Bishop, and John M. Carroll. Experiences in cyber security education: The mit lincoln laboratory capture-the-flag exercise. In *Proceedings of the 5th USENIX Workshop on Cyber Security Experimentation and Test (CSET 2012)*. USENIX Association, 2012. Accessed: 2025-12-07.

[3] Kevin Bock, George Hughey, and Dave Levin. King of the Hill: A novel cybersecurity competition for Teaching Penetration Testing. 1 2018.