Haaukins: A Highly Accessible and Automated Virtualization Platform for Security Education

Thomas Kobber Panum*, Kaspar Hageman*, Jens Myrup Pedersen*, René Rydhof Hansen†
*Department of Electronic Systems, †Department of Computer Science
Aalborg University, Denmark
Email: tkp@es.aau.dk, kh@es.aau.dk, jens@es.aau.dk, rrh@cs.aau.dk

***Abstract*—Education of IT security can include a tedious and frustrating experience for novice students and organizers. We have sought out to create an education platform that improves upon this experience, through automation, and individualized learning labs. These learning labs hosts are isolated clusters of virtual computer instances, representing real and insecure computer networks. The platform, named Haaukins, improves upon typical accessibility issues of students and cumbersome configuration management for organizers. In order make the platform accessible for other organizations, it has been open sourced.**

***Keywords*—virtualization platform, teaching, information security, capture-the-flag**

I. Introduction

On a global scale, there is an increasing demand for qualified personnel for information security related positions. The $(ISC)^2$ estimates the shortage of security professionals to be 2.93 million persons [1]. The Danish government has recognized this gap and is increasing its funding of cyber security education and research in response [2]. In parallel, Aalborg University (Denmark) has started to address this issue by adding cyber security to the curriculum of relevant engineering educations.

This effort is accompanied by the university's ambassador programme, which engages university students to reach out and teach high school students about their respective field of education. For cyber security, this involves running introductory workshops with subjects such as vulnerability scanning and exploitation. The sessions are typically short, between one and two hours of duration, and are initiated by a short lecture followed by exercises where participants interact with a computer system, referred to as a *lab*. As an organizer, managing these labs is time-consuming and error-prone, especially as participants often need assistance, e.g., to gain access to the lab. Our collective experience from hosting sessions with existing solutions led to a series of requirements for an education platform.

In order to have a solution that conforms to these requirements, we have developed a new virtualization platform, Haaukins, which grew from the collaboration project 'Danish Cyber Security Clusters'. The platform differs from existing work by automating the tedious management of the labs, while also providing an individualized experience to improve learning and making labs accessible with no configuration. It is designed to support the common exercise format, Jeopardy capture-the-flag (CTF) [3], in which participants must gain access to certain secret information contained within a given computer system.

II. Related Work

Although there exist numerous platforms for deploying labs of connected, virtual instances, none of these fulfill the requirements of our specific use case in its entirety. The popularity of CTFs as a competition format has resulted in a range of commercial, closed platforms to support running such events [4], [5]. The fact that these platforms are not open makes them unsuitable for an educational use, since the educator is completely reliant on the owner in both the access to the platform and the material hosted on them.

A range of existing platforms places various teams in a single, shared virtual network. The motivations for doing so range from a performance consideration [6] to the desire to host attack and defense CTFs (ADCTFs) [7], [8]. In ADCTFs, participants not only attempt to hack other machines, but actively have to protect their own against others. Given the limited prior knowledge of our target audience, it is infeasible for us to host ADCTFs.

PicoCTF is organized yearly and — similarly to Haaukins — aims to create an interest for information security among high school students [9]. The platform is developed as an interactive game, and thereby does not reflect a realistic scenario.

In summary, the related work is either closed, not suitable for our CTF format or does not represent the real world in a realistic fashion.

III. Design Goals

The design of Haaukins has primarily been driven by four design goals: automation, transparency, accessibility and realism. In the following subsections, we describe each design goal in some detail.

a) Fully Automated (DG_{FA}): In the process of preparing a CTF, numerous components need to be instantiated, configured and connected correctly. Manual configuration of labs is time-consuming process and errors can have severe consequences, e.g. a virtual instance being completely unreachable from one wrongly assigned IP address. Haaukins reduces the preparation time for events through full automation of the configuration of its labs.

b) Transparent (DG_T*):* Discovering and identifying a security vulnerability requires investigation and exploration. This element of exploration must be contained in Haaukins, without causing participants to end up in dead ends that can hinder their learning [10], [11]. If the design of the exercises is the cause of these dead ends, it has be identifiable. Ideally a platform allows for gaining insight into this, by providing a method for observing participants' behavior and make them available for further analysis.

c) Highly Accessible (DG_{HA}*):* In the setting of a one- or two hour lecture, time is a valuable asset that should not be wasted on irrelevant aspects. Given the short timespan, the overhead of accessing a virtual lab can take up a significant fraction of the allocated time. Beginners might find this non-trivial and it may be a hurdle for progressing. We strive to minimize the overhead, and ideally want participants to access their virtual lab in a matter of minutes, independently from their physical location and the operating system they use.

d) Realistic (DG_R*):* For skills learned through a simulated environment to be valuable, they must be transferable to the real world setting. Haaukins strives to do so by ensuring that the designed labs are realistic replicas of real vulnerable computers and their networks. The labs are interacted with using a professional toolkit, that continuously evolves to remain relevant for trends in security vulnerabilities. Experiences gained from the labs should indistinguishable from real world settings, and exercise developers should not be restricted by the limitations of the platform.

IV. Overview

A key feature of Haaukins is the option for multiple organizers to host simultaneous sessions (referred to as EVENTS) with one or more exercises for participants. For an event, a group of participants registers as a TEAM which is assigned an environment upon registration. This environment is referred to as a LAB and is represented by a network of virtual INSTANCES. Teams are tasked to discover unique identifiers (or FLAGS), that function as proof for solving a CHALLENGE, which can be checked through a web application, and upon being valid, are counted as positive scores for the team.

A new event starts with any composition of EXERCISES and FRONTENDS (see Figure 1). A frontend is an instance that a team gets to control via a graphical user interface. An exercise is composed of any number of IMAGES, which are templates from which an instance is created. The specification of an image consists of: a virtual disk image, any number of records, and any number of challenges. The virtual disk image can either be a Docker image or an Open Virtual Appliance (OVA) package, e.g. *nginx* or *kali.ova* in Figure 1. The records are DNS records, which map domain names to IP addresses per lab, and are necessary for the communication among instances.

The instantiation of a lab consists of automatic creation and configuration of instances based on the collective specification of exercises and frontends. In addition to the associated instances, every lab also contains a set of core services to support connectivity and service discovery for instances, these services are DHCP and DNS respectively. The instantiation process also involves inserting unique flags in instances and randomizing IP address ranges, thereby individualizing each lab and its challenges.

V. Design

Haaukins consists of a client and a server component, `hkn` and `hknd` respectively, enabling multiple organizers to interact with the same instance of `hknd` independently of each other.

The daemon process, `hknd`, controls the life cycle of internal data structures and the orchestration of all components; it further serves as an application wide reverse HTTP proxy acting as a single point of entry for all the web traffic that comes in from the participants of the platform, and redirects the traffic to the correct virtual instances. On an event-level, there are two third-party components being managed: CTFd [12] and Guacamole [13]. CTFd is a web application responsible for the graphical user interface for the participants, which allows them to access their respective event through a web browser. Through this interface, the teams can view their respective exercises, fill in the results for their respective challenges, and are directed to their respective frontends which are accessible through Guacamole. Guacamole is a web application which allows for streaming remote desktops to a web browser, and is more thoroughly covered in Section V-0c.

The client, `hkn`, provides a command-line interface (CLI) that allows organizers to interact with `hknd`, e.g., to create events, listing exercises or resetting instances for certain teams.

a) Automated Orchestration: Haaukins uses Docker containers and Oracle VirtualBox (VirtualBox) virtual machines for deploying multiple isolated services within labs. These technologies are both used to allocate and isolate the resources (e.g. CPU and RAM) of a physical machine into virtual instances, but their ability to do so differs in terms of computation overhead and level of isolation.

All instances in a lab are connected to the same virtual network, to ensure that all instances can communicate among each other. Concretely, Docker Macvlan is used for the networks, resulting in a LAN network topology that allows for promiscuous network monitoring and gives participants the ability to observe the entire network traffic. External network access, such as the Internet, has been disabled to prevent participants from abusing the instances.

b) Lab Individualization: In highly competitive CTFs, the sought-after flags are identical across participating teams, since the competitive nature is an incentive for keeping found flags private. From our experience this incentive does not transfer to an educational setting, as students are more inclined to share solutions, causing cheating that negatively influences DG_T through false progress. To combat this issue, Haaukins individualizes each lab through two techniques: dynamic flags and dynamic subnets. Creating dynamic flags is the process of creating unique flags on a per team basis in dynamic exercises, and thereby prevent the sharing of flags. Implementing dynamic subnets involves hosting labs on networks with randomized private IP ranges, which is a significant

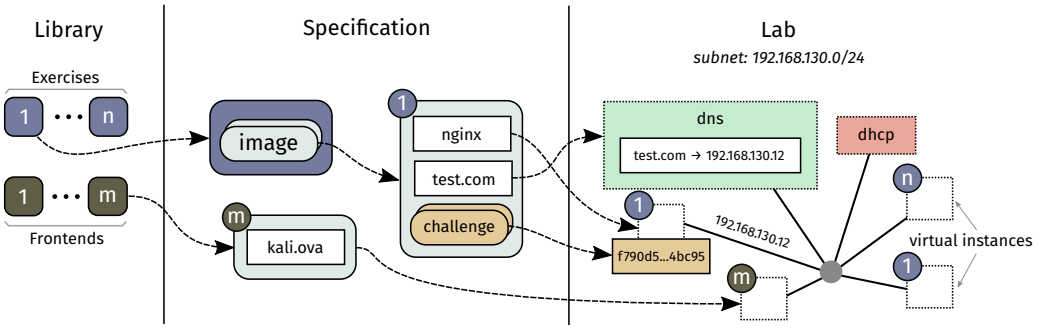

Fig. 1: Levels of abstraction of a lab. Based on a selection of exercises and frontends (*left*), Haaukins uses their specification (*middle*) to create an individualized lab of instances per team (*right*).

mutation in the setting of network analysis exercises. This prevents students from sharing information about IP addresses of instances, with only a few exceptions of core services, i.e., DNS and DHCPD.

c) Accessible Through a Web Browser: Based on our experience, novice participants often encounter problems with the use of a (desktop-)client for remote access, such as VPN or RDP. These problems led to the choice of using Apache Guacamole within Haaukins, which is a web application that allows for accessing remote desktop protocols, e.g. RDP, through a modern web browser. Guacamole uses a backend daemon for translating standardized protocols to WebSocket traffic that is interpretable by a JavaScript client in the user's browser. Within Haaukins, the built-in capabilities of VirtualBox is utilized for creating RDP access to frontends. This access is then translated by Guacamole in order to be accessed from the participant's browser, requiring no installation process and being accessible from any physical location.

d) Monitoring Participants: The design of effective exercises is complicated, as the exercises need to be open-ended in order to adhere to DG_R, while ensuring the openness does not cause participants to become stuck. In Haaukins, participants are initially tasked with identifying vulnerabilities, before actively exploiting them. If the participants become stuck at this stage, it will hinder their ability to learn from the exercises. To determine when and how this behavior occurs, Haaukins has the ability to monitor and log participant interaction with the platform. Only if consent is granted, the WebSocket traffic of Guacamole is captured using the reverse proxy of `hknd` and transformed to a stream of key presses. These streams can then be further analyzed in order to determine participant behavior, e.g. programming activities and terminal usage, that can potentially influence changes to the teaching material.

VI. Conclusions

We present a novel education platform, Haaukins, that differentiates itself from existing CTF platforms by having improved accessibility, full automation, observability of participant behavior and high degree of realism. The platform presents itself as a web application that provides highly accessible lab environments for participants, accessible within minutes without prior experience. It completely automates the creation, configuration, teardown of all its components. Each lab is personalized through unique mutations of flags and IP addresses, which discourages cheating among participants. Since the labs in Haaukins are designed to be a realistic representation of a realistic network, learnings from the platform translate directly to real-world scenarios. In order to support other education institutions in conducting short CTF workshops, the platform is available as an open source project on GitHub[1] with a GNU GPLv3 license.

References

[1] (ISC)^2, "Cybersecurity Workforce Study." [Online]. Available: https://www.isc2.org/research/workforce-study

[2] Danish Ministry of Finance, "Danish Cyber and Information Security Strategy 2018-2021." [Online]. Available: https://uk.fm.dk/publications/2018/danish-cyber-and-information-security-strategy

[3] T. Chothia and C. Novakovic, "An Offline Capture The Flag-Style Virtual Machine and an Assessment of its Value for Cybersecurity Education," *2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15)*, 2015.

[4] Hacking-Lab, "Hacking-lab." [Online]. Available: https://www.hacking-lab.com/

[5] Hack The Box Ltd, "HackTheBox.eu." [Online]. Available: https://www.hackthebox.eu/

[6] L. McDaniel, E. Talvi, and B. Hay, "Capture the Flag as Cyber Security Introduction," in *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE, 2016, pp. 5479–5486.

[7] G. Vigna, K. Borgolte, J. Corbetta, A. Doupe, Y. Fratantonio, L. Invernizzi, D. Kirat, and Y. Shoshitaishvili, "Ten Years of iCTF: The Good, The Bad, and The Ugly," *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, pp. 1–7, 2014.

[8] J. Mirkovic and P. A. H. Peterson, "Class Capture-the-Flag Exercises," in *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, 2014.

[9] P. Chapman, J. Burket, and D. Brumley, "PicoCTF: A Game-Based Computer Security Competition for High School Students," *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, no. May, 2014.

[10] W.-C. Feng, "A Scaffolded, Metamorphic CTF for Reverse Engineering," *2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 15)*, 2015.

[11] K. Chung and J. Cohen, "Learning Obstacles in the Capture The Flag Model," in *2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14)*, 2014.

[12] CTFd Maintainers, "Ctfd." [Online]. Available: https://github.com/CTFd/CTFd

[13] The Apache Software Foundation, "Apache guacamole." [Online]. Available: https://guacamole.apache.org

[1] https://github.com/aau-network-security/haaukins