Engaging Students with Personalized and Remotely Orchestrated Cybersecurity Training Exercises

Vojdan Kjorveziroski*
Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje
Email: *vojdan.kjorveziroski@finki.ukim.mk

***Abstract*—Hands-on cybersecurity exercises offer students a glimpse into the real world, and through increased engagement improve their knowledge retention. This paper describes a virtual environment which allows educators to create personalized exercises at scale, encouraging students to apply learned concepts in practice. By introducing gamification aspects, and promoting teamwork, a competitive environment is created, further enticing users to participate. Students' achievement results show that such an environment aids their learning process, a fact also backed by a feedback survey performed at the end of a network security course, where the described environment was first utilized.**

***Index Terms*—cybersecurity education, training, virtual environment, cybersecurity exercises**

I. Introduction

The ever-increasing number of security related breaches [1], coupled with an enormous workforce gap [2], stresses the importance of including cybersecurity training in higher education. To maximize the effectiveness of such training, it should be applied not only to security related university programmes, but across the wider IT curricula [3]. In this way, even students that might not work primarily in cybersecurity related roles will be included, giving them a solid understanding of the omni-present cyberthreats and means of avoiding them. One of the most popular methods of teaching cybersecurity is through hands-on exercises, which has been shown to be beneficial to the students' knowledge retention rate and scientific achievement [4][5]. However, such exercises require large hardware resources which might not be at every student's disposal. The recent Covid-19 pandemic and the move towards online lectures, has only exasperated the problem, since students no longer have access to the laboratories within their university campuses.

To overcome these obstacles, we have created a virtual laboratory that was used by students attending a network security course at our faculty. The purpose of this virtual environment is twofold. Firstly, it serves as a playground where the students work on their weekly laboratory exercises, using personalized virtual machines. Secondly, it allows us to organize an end-of-semester red team/blue team project where each team has the opportunity to both defend their own infrastructure, as well as attack and exploit the virtual machines of the other teams. Through these incremental laboratory exercises, together with their extensive preparatory material [6][7], students are introduced with various software tools that they can use independently during their final course project, for which only basic guidelines are provided, thus encouraging teamwork and mutual collaboration.

The aim of this paper is to describe the chosen architecture of the virtual laboratory, share details about the exercises, and report on feedback received by the participating students of the course. The rest of the paper is structured as follows: in section two we present related work relevant to the topic; in section three we provide details about the implementation of the virtual environment, describing its architecture, and report on the exercises used during the course; in section four we share the results of students' feedback that we have received as part of an end-of-semester questionnaire, as well as encountered hurdles. We conclude the paper with section five, outlining future plans and improvements to the platform.

II. Related Work

There are numerous contributions to the topic of virtual laboratories for education and training purposes. Generally, in terms of the used infrastructure, the implementation is done in one of two ways: either by employing the scalability and ease-of-use of the cloud, or by using desktop machines or local dedicated servers, thus increasing control, isolation, and user privacy.

EDURange [8][9], DeterLab [10] and [11] are examples of cloud based virtual environments implementing various scenarios and challenges on different topics related to cybersecurity. The main problems with this cloud based approach are terms of service breaches, as well as monetary cost. Many cloud providers explicitly forbid activities that might pose a risk to the service or any third-party [12]. Furthermore, associated costs can reach hundreds of dollars even for very small groups of students [11], which might be prohibitively expensive for certain institutions.

A slight variation of the cloud concept are publicly accessible cybersecurity training sites such as TryHackMe[1], and HackTheBox[2], which allow anyone to access an isolated virtual environment after the registration procedure is completed. Unfortunately, in these platforms mutual user collaboration is non-existent, since multiple users cannot share a single virtual machine, or access a group of machines together. Furthermore, the scenarios are generic, barring any user personalization.

[1]TryHackMe - https://tryhackme.com/ (Last accessed: 05.03.2021)
[2]HackTheBox - https://www.hackthebox.eu/ (Last accessed: 05.03.2021)

The on-premise approach overcomes these challenges, presuming that there is available equipment for its implementation, but more importantly, that the necessary time is dedicated to environment setup, along with scenario customization. The authors of [13] and [14] have both successfully taken this approach. However, in the first scenario, remote access is provided through a remote desktop protocol, limiting the choice of tools that can be used during the exercises, while in the second, access to the platform is possible only through other laboratory workstations located on-premise.

Recent advancements in orchestration tools, as well as the widespread availability of open-source virtual private network (VPN) software, has motivated us to create a virtual laboratory that will be capable of running personalized virtual machines for each user, providing as seamless access as possible, extending the network through virtual tunnels to the remote workstations of the students.

III. Implementation

As a result of the Covid-19 pandemic, our faculty has completely shifted to remote classes, without any requirements for physical presence by the students. Even though the adoption of online teaching methods was completed successfully, challenges relating to the individual weekly laboratory exercises have arisen. These exercises are of great importance to the students and allow them to apply the concepts learnt during the lectures in practice. Most of these exercises have previously taken place in the faculty's computer laboratories, where the available hardware met the high requirements of the used software. However, due to Covid-19 measures, the computer laboratories are no longer accessible, leaving the students with no other option but to perform all laboratory exercises locally, often on subomtial hardware, and without interaction either with the educators or with their colleagues.

In order to mitigate these issues, we have decided to implement a virtual environment on campus, repurposing some of the now unused machines from the computer laboratories. This environment would be remotely accessible by the students, and apart from providing the necessary computing capacity for the individual exercises, it would also allow us to implement interactive and collaborative challenges, encouraging teamwork. Even though we are aware that the setting up of such an environment requires high initial effort, both in terms of infrastructure configuration and security, we recognize that this could also be exploited towards educational purposes. Many infrastructure and security courses deal with such topics and the architecture of this virtual laboratory would be the perfect opportunity to showcase a production-like environment to the students, along with all of its different components.

Contributing to these goals, we have identified five formal objectives that we would like to meet and that are essential to the virtual environment: 1) network isolation with strict monitoring and limited internet access for vulnerable machines; 2) versatility and scalability of the environment; 3) ease of administration; 4) ease of use and seamless remote access for students; 5) use of latest technologies and best security practices [15].

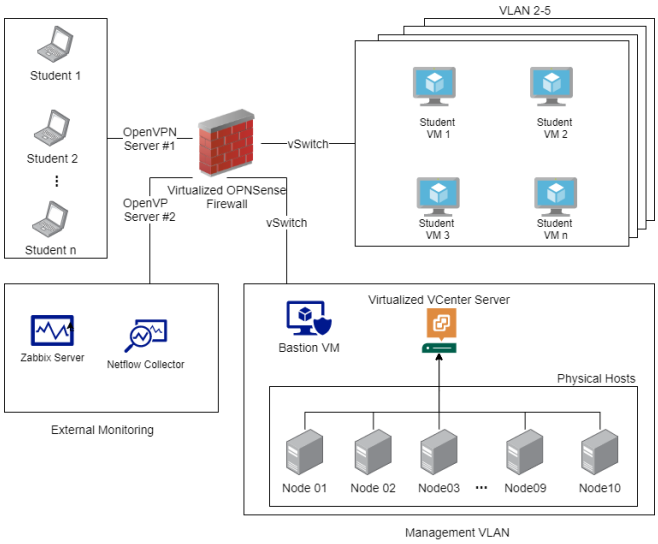

Figure 1: Network diagram for the virtual environment

A. Environment Setup

Figure 1 shows a graphical representation of the network topology of the completed environment. In the subsections that follow we provide detailed explanation for the reasoning behind each decision and how it has contributed to the meeting of the previously outlined objectives.

1) Network Isolation with strict monitoring and limited internet access for vulnerable machines: Sharing computing resources among different students, especially when known vulnerable software is being run, requires strict network isolation from the production network, ensuring its integrity. To this effect, extensive network security measures have been implemented. All of the physical hosts used in our virtual environment are connected to a manageable switch on which dedicated virtual local area networks (VLANs) are configured, completely separate from the regular campus network. A single VLAN is dedicated to management, and it is used both for interconnectivity between the hosts themselves, as well as for a bastion virtual machine, used by the educators to manage the environment and provision new virtual machines for the exercises. The instantiated virtual machines used by the students are placed in the remaining VLANs. More than one VLAN is required for this purpose, since some exercises require two or more VMs per student and granular firewall rules are needed to limit their connectivity, making the scenario more realistic.

Internet access is provided by an OPNSense firewall appliance[3], running virtualized on one of the compute nodes. Even though there are multiple open-source firewalls that could be used for this purpose, we have selected OPNSense because of its application programming interface (API), allowing us to programatically manage multiple aspects, without accessing

[3]OPNSense - https://opnsense.org/ (Last accessed: 05.03.2021)

the web interface. In this way we can easily restrict internet access to the virtual machines, and configure the necessary rules using the same orchestration tools for machine deployment. Depending on the scenario, either no internet connectivity is possible from the students' virtual machines, or access only to particular destinations is allowed, for example package mirrors for downloading new software from a distribution's official repositories.

The status of both the physical hosts, as well as the supporting virtual machines such as the bastion and the firewall appliance shown in Figure 1, is monitored at all times using the open-source Zabbix[4] monitoring system. A dedicated Zabbix instance is installed remotely, on the faculty's cloud. Automatic discovery is done of all student virtual machines and alerts are set up to detect any abuse of the infrastructure, such as very high CPU usage for long periods of time, or high disk utilization. This monitoring setup is also reused during the final team projects, where network checks are configured to validate the availability of each VM, and conformity to the established rules.

Traffic between the virtual machines as well as to the public Internet is monitored using a NetFlow collector, which is also installed in the faculty's cloud. As a result of this setup, detailed information about each traffic flow is available, along with aggregate statistics such as traffic volume, most common destinations, and top sources, allowing us to detect anomalous behavior.

2) Versatility and scalability of the environment: Even though the presented virtual environment was first used for a Network Security course as a sandbox for laboratory exercises, the idea is to make it as versatile as possible, giving the possibility to be reused in the future, for different courses. For this reason, we have selected ESXi as the hypervisor to be installed on each physical host, and to achieve central management, a virtualized instance of VCenter Server is deployed. We have opted for a VMware environment because of the extensive API provided to programatically manage it, as well as the availability of open-source tools to automate the provisioning of new virtual machines. Of course, other open-source alternatives can be used too, with notable examples being Proxmox [5] and XCP-ng [6].

Using the chosen architecture, it will be possible to expand the computing capacity available to the virtual environment in the future, by simply adding additional physical machines to the 10 already present, and joining them to the same compute cluster, thus ensuring its scalability.

3) Ease of administration: During the software selection process great care was taken to ensure that all of the chosen components offer well defined APIs, or at least can be extended to do so, allowing us to completely automate both the creation of the VMs as well as the associated firewall rules, limiting the VMs' network connectivity. Unfortunately, even though OPNSense does have an API that can be used, not all provided features have been covered by it yet. As a result, we have created a Python library that emulates the HTTP requests sent when configuration is done via a browser, allowing us to completely automate both the user provisioning process, VPN enrollment, and rule creation – something not entirely possible through the API itself. For the VM provisioning, two different approaches are used, depending on the type of the scenario. In cases where a basic virtual machine is needed, with minimal additional configuration or preinstalled software, the Ansible orchestration tool is utilized, which uses the VCenter API to instantiate a new virtual machine for each enrolled student. Alternatively, exploitable virtual machines with known vulnerable software installed are created using the open-source SecGen [16] tool, which has support for VMware environments, and allows automatic provisioning as well. More details about the usage of SecGen and the vulnerable virtual machines that it helped deploy are given in the subsection Scenarios. This approach can be very easily adapted to work in different contexts and create virtual machines for other courses.

4) Ease of use and seamless remote access for students: Even though majority of the course participants were in their third year of studies, there were also instances of younger students, so no prior knowledge with virtual machines or remote access technologies was assumed. For this reason, the connection to the virtual environment had to be as easy as possible, without burdening the users with infrastructure specifics. This approach allows us to introduce this environment to even younger students in the future as well. In terms of grading and instruction distribution, we opted to integrate with existing and well established faculty systems, mainly the Moodle learning management system (LMS) which is used across all courses at the faculty, and is familiar to every student.

Remote access to the virtual environment is made possible through a VPN server running on the virtualized OPNSense firewall. Individual profiles are created for all enrolled students, and the configuration file together with the private and public keys required for authentication to the VPN are uploaded as a personal feedback to a Moodle activity for each user. With this approach, it is ensured that students can only access their own VPN profile, without the possibility of impersonation, as well as provide the means to redownload the profile directly from the LMS in case it is lost. Each user is assigned a known static IP addresses within the VPN subnet, allowing us to automatically create the necessary firewall rules whenever a new exercise is published. The choice for a VPN solution instead of some other remote desktop approach was done so that students could install attack tools and penetration testing distributions locally on their own computers, and then use these to attack the vulnerable machines provided as part of the exercises.

5) Use of latest technologies and security best practices: The elaborate network setup described previously, apart from ensuring the integrity and security of the platform itself, fulfills another purpose as well. During the Network Security course

[4]Zabbix - https://www.zabbix.com/ (Last accessed: 05.03.2021)
[5]Proxmox - https://www.proxmox.com/en/ (Last accessed: 05.03.2021)
[6]XCP-ng - https://xcp-ng.org/ (Last accessed: 05.03.2021)

we cover topics related to firewalls, intrusion detection systems (IDS), intrusion prevention systems (IPS), network monitoring, and automation. This infrastructure allows us to demonstrate such systems in practice, by showcasing them in the context of the virtual environment. This has a positive effect on the students, since they incrementally learn about the inner-workings of a system that they have already had experience with, allowing them to easily grasp the concepts behind each tool. It also servers as an inspiration regarding what tools and practices can be used for their final course assignment.

B. Scenarios

During the course, there are weekly laboratory exercises which gradually introduce students to practical tools related to the topics covered in the lectures. All of these exercises also directly contribute to the students' ability to complete the final project, outfitting them with the necessary knowledge to individually install and configure complex defensive and offensive systems.

The creation of the virtual machines used during these activities is performed by the educators using the open-source SecGen tool, which comes with a wide range of preconfigured scenarios containing different types of vulnerable software, ready for exploitation. Each vulnerability is represented by a so called module, and new modules can be manually created by providing the source files for a vulnerable software whose installation and setup can be done by the Puppet orchestration tool, natively supported by SecGen. Multiple modules comprise a single scenario, which can be customized per user with unique system parameters and flags, which can then be used for validating the challenge solution, and limiting plagiarism. Unfortunately, even though there is a large library of existing scenarios, some of them are not fully functional due to software deprecation or unreachable download locations. As a result, we have created our own scenarios both for the laboratory exercices and the final project, by reusing some of the existing SecGen modules, and extending them by using other orchestration tools, thus adding additional features.

Table I: Laboratory exercise description

Name	Virtual	VMs No.	Duration
Environment setup	✓	0	1 week
Password cracking	✓	1	1 week
SELinux	X	1	end of semester
WireGuard setup	✓	2	2 weeks
CloneZilla backup and restore	X	2	end of semester
Metasploit exploitation	✓	1	1 week

1) Laboratory Exercises: An overview of the laboratory exercises is given in Table I, providing details about the covered topics, number of virtual machines, and deadlines. Two of the exercises, SELinux and CloneZilla, are intentionally done locally by the students, to better introduce themselves with virtual machine deployment and virtualization software. Both of these exercises have minimal hardware requirements, and could be completed on computers with even less than 4GB of memory, so none of the course participants should have any difficulty in terms of hardware performance. Familiarity with host based virtualization tools would also allow students to install penetration testing distributions locally, enabling them to explore additional software on their own. Additionally, by using the provided VPN profiles, they can utilize their own hardware as an extension to the virtual laboratory. In this way, elaborate monitoring setups can be set up for the final course project.

The laboratory exercises gradually increase in their complexity, and assume no advanced knowledge in either system administration or use of command line tools. The aim of the first exercises is to introduce the students to the concept of virtual private networks, and to help them establish an initial connection to the virtual laboratory using their personal VPN profile. It is crucial that every student can seamlessly connect, since this is a prerequisite for the completion of the other exercises as well as the final project. In our case, no major problems were encountered, any by the second exercise, all students were successfully connecting to the virtual environment either from their host operating systems, or from penetration testing distributions running in virtual machines on their own systems.

The second exercise practically demonstrates the drawbacks of weak passwords, and how they can be cracked using the right tools in short amounts of time. This is also the first time that students are introduced to the concept of capture the flag challenges, where they need to obtain a unique string called a flag for each exploited vulnerability, and submit this flag for automatic verification via the Moodle platform. In this way, by utilizing gamification through the concept of flags that can be exchanged for points, the exercises are made more interesting for the participants. It also introduces a whole new competitive aspect, especially during the final projects, where teams compete with one another for the capture of each others' flags.

The SELinux and Metasploit exercises aim to present robust tools for defense, and offense, respectively, both being widely used in the real world. These tools also underpin many of the students' strategies during the final project.

Even though by the fourth exercise all of the students are using a VPN regularly, its aim is to explain the underlying technology that they have utilized during the previous laboratory exercises, by walking them through the setup process of such an infrastructure from scratch. Finally, the use of standardized images and image-based backups is promoted as means for faster provisioning of new machines, as well as a recovery mechanism after an attack. Having a backup plan is always important, and this topic fits nicely among the other offensive and defensive techniques presented to the students.

2) Final Project: In contrast to the well documented laboratory exercises, the final project is a team activity, where students can choose their teammates by themselves. Each team is provided with a single virtual machine containing multiple vulnerabilities that can be chained together to acquire root access. No additional information is provided to the teams, apart from basic information on how to connect to their VM.

In order to provide a level playing field among the teams, two distinct phases are organized, where each team has the opportunity to both showcase their offensive skills, attacking the virtual machines of the other colleagues, and improve their defensive skills, by preventing exploitation of their own infrastructure. During the first, blue, phase each student has access only to the their team's virtual machine, and this time is used to survey the installed software, and devise an action plan for defense. Students are encouraged to collaborate by themselves, and to reuse the knowledge gathered during the laboratory exercises, setting up additional tools that might prevent unauthorized access by other users. During the red phase, all network access restrictions are lifted, and the environment becomes a free-for-all tournament between the teams, each one trying to exploit as many vulnerabilities as possible, thus capturing unique flags from their counterparts.

During both phases, users are empowered to come up with comprehensive plans which they outline in their final report. There are no restrictions to the tools and methods employed, as long as: i) users do not patch, uninstall or in any other way make inaccessible the vulnerable software running on their machine; ii) purposely execute denial of service attacks; iii) alter the flags on their virtual machines; iv) define firewall rules resulting in blanket traffic restrictions for whole network ranges. Instead, users are encouraged to employ extensive traffic monitoring to their VMs, and try to distinguish malicious from regular traffic to the installed services, selectively blocking it. Log deletion, along with other actions such as banner changes that can be used to either cover the attackers' tracks or show off that a victim's machine has been compromised, are allowed, as long as it does not impact the original functionality of the server. This provides bragging rights to the teams, making the whole project even more competitive and personal.

Monitoring of the vulnerable services deployed on each VM is performed during the whole duration of the project, with immediate notifications sent to the educators, informing them of any rule breaches. Repeating offenders are disqualified.

IV. Results

This was the first time that an on-premise virtual laboratory was created for the students of the Network Security course at our faculty. It allowed us to implement interactive and personalized cybersecurity challenges, while at the same time relieving students from running resource intensive virtual machines on their own computers. This had a great impact on the level of involvement in both the laboratory exercises and the final end of year project. The usefulness of the proposed approach is further confirmed by the results of the questionnaire that students have been asked to fill at the end of the semester. Four of the included questions were directly related to the organization of the laboratory exercises and the final project:

1) I found the laboratory exercises interesting, and they helped me learn new things. (Five point Likert scale)
2) What is your preferred deadline for the laboratory exercises? (Individual deadlines for each exercise; End of semester, for all exercises; No preference)
3) What is your opinion on the virtual environment? (It appropriate, providing great working conditions; It was complex, and hindered my course activities; I did not use the virtual environment; No opinion)
4) I found the final project interesting, and it helped me learn new things (Five point Likert scale)

The course had a total of 63 participants, 36 of which have completed the questionnaire, accounting for more than half of all enrolled students. Keeping in mind that the course is not yet officially finished, and there are additional opportunities to complete the final exam, users are still able to submit feedback. In the remaining of this section we present the results, noting that the given percentages represent the fraction of the students that have responded to that particular question, instead of all users who have taken the questionnaire. None of the questions were mandatory, so some of them have less than 36 responses. As a result, not all percentages might add up to 100.

Figure 2 shows the results of questions 1 and 4, for both of which a five point Likert scale was used. The first question was answered by all 36 respondents, while the fourth by 33. Positive sentiment towards both the organization of the laboratory exercises and the final project is evident, with 66.67 per cent and 54.54 per cent of the respondents, respectively, completely agreeing with the statement. However, contrary to our assumptions, 66.67 per cent of the users who have submitted an answer to the second question have stated that they prefer an individual deadline for each exercise that they had to do on their own computers, instead of a single end-of-semester deadline, allowing them to finish the exercises at their own pace. On the third question, regarding the virtual environment, of the 35 responses, 30 (83.33 per cent) have stated that it was appropriate, providing great working conditions, 3 (8.33 per cent) that they found it too complex, and 2 (5.56 per cent) did not use the environment at all.

For the duration of the whole semester, the hardware resources contributed by the 10 bare metal machines taking part in the cluster were sufficient for all exercises, and no major issues were encountered. Power was cut twice to the location where the machines were hosted, but this event was taken into account during the initial setup, and automatic power-on was enabled both of the physical machines, and all virtual machines, so no manual intervention was needed. During the final project, two rule violations using the monitoring setup were detected and the team members were officially warned, after which no more rule breaches occurred.

The submitted reports by each team showed that 6 out of the 10 participating teams have successfully exploited at least one other virtual machine, and 2 of the remaining 4 have at least successfully detected no less than one exploitation attempt. What is very encouraging is that the majority of the participants have successfully applied the concepts described during both the lectures and laboratory exercises and have come up with elaborate defensive mechanisms, including

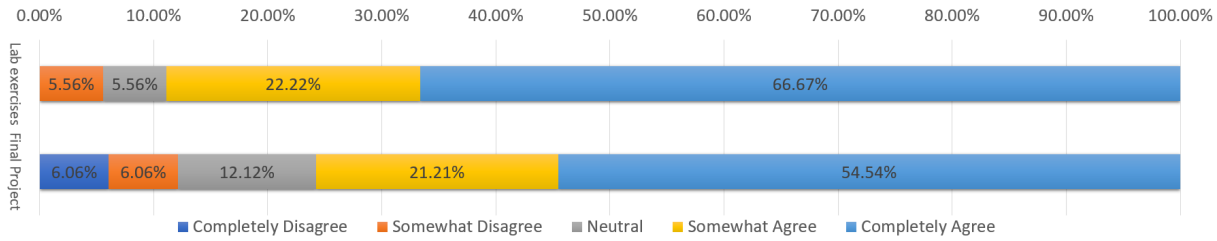

Figure 2: Responses to questions one and four of the feedback questionnaire

installation of intrusion detection systems and log monitoring, thus justifying the approach taken during the semester.

V. Conclusion

We have successfully implemented a virtual environment that is being used for cybersecurity exercises during the Network Security course at our faculty. By repurposing existing physical machines no longer utilized as a result of the Covid-19 pandemic and the associated closure of the computer laboratories on campus, we have created a scalable infrastructure, ready to be reused in future courses as well. This approach provides a more interactive learning experience for all students, while also overcoming compute capacity issues for students with older hardware. Using automation and orchestration tools, the provisioning time has been reduced to a minimum, making it a viable option for courses with large numbers of participants. The extensive network monitoring and traffic filtering performed using open-source tools ensures that the virtual environment is completely isolated from the main faculty network, thus providing a sandbox where even vulnerable software can be run without impact to external systems. Access using widely available open-source VPN software makes it possible for students to access this virtual infrastructure from any device in their possession, and at the same time giving the possibility of reusing the environment in different contexts, not solely for cybersecurity exercises.

In the future we plan the final project reports to be accessible to the educators at all times, providing the possibility to individually track each teams progress, and once the assignment is completed, share the reports among the other participants. In this way, each team can reflect on their own contribution, and how their actions were seen through the eyes of the other teams, providing a whole new aspect to the activity. Furthermore, we also plan to introduce even more interactive exercises, such as a king of the hill scenario [17], where students are asked to progress with capturing and then defending a set of virtual machines.

References

[1] "Cyber Security Breaches Survey 2020." [Online]. Available: https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2020/cyber-security-breaches-survey-2020

[2] "Cybersecurity Workforce Study 2020," ISC2, Tech. Rep. [Online]. Available: https://www.isc2.org/-/media/ISC2/Research/2020/Workforce-Study/ISC2ResearchDrivenWhitepaperFINAL.ashx

[3] B. Lunt, J. Ekstrom, S. Gorka, G. Hislop, R. Kamali, E. Lawson, R. LeBlanc, J. Miller, and H. Reichgelt, "Curriculum Guidelines for Undergraduate Degree Programs in Information Technology," Association for Computing Machinery, New York, NY, USA, Technical Report, 2008.

[4] D. C. Rowe, B. M. Lunt, and J. J. Ekstrom, "The role of cybersecurity in information technology education," in *Proceedings of the 2011 conference on Information technology education*, ser. SIGITE '11. New York, NY, USA: Association for Computing Machinery, Oct. 2011, pp. 113–122.

[5] P. M. Stohr-Hunt, "An analysis of frequency of hands-on experience and science achievement," *Journal of Research in Science Teaching*, vol. 33, no. 1, pp. 101–109, 1996.

[6] R. Weiss, F. Turbak, J. Mache, E. Nilsen, and M. E. Locasto, "Finding the Balance Between Guidance and Independence in Cybersecurity Exercises," 2016. [Online]. Available: https://www.usenix.org/conference/ase16/workshop-program/presentation/weiss

[7] P. A. Kirschner, J. Sweller, and R. E. Clark, "Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching," *Educational Psychologist*, vol. 41, no. 2, pp. 75–86, Jun. 2006.

[8] R. Weiss, F. Turbak, J. Mache, and M. E. Locasto, "Cybersecurity Education and Assessment in EDURange," *IEEE Security & Privacy*, vol. 15, no. 3, pp. 90–95, 2017.

[9] R. S. Weiss, S. Boesen, J. F. Sullivan, M. E. Locasto, J. Mache, and E. Nilsen, "Teaching Cybersecurity Analysis Skills in the Cloud," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '15. New York, NY, USA: Association for Computing Machinery, Feb. 2015, pp. 332–337.

[10] J. Mirkovic and T. Benzel, "Teaching Cybersecurity with DeterLab," *IEEE Security & Privacy Magazine*, vol. 10, no. 1, pp. 73–76, Jan. 2012.

[11] K. Salah, M. Hammoud, and S. Zeadally, "Teaching Cybersecurity Using the Cloud," *IEEE Transactions on Learning Technologies*, vol. 8, no. 4, pp. 383–392, Oct. 2015.

[12] "AWS Customer Agreement." [Online]. Available: https://aws.amazon.com/agreement/

[13] C. Willems and C. Meinel, "Tele-Lab IT-Security: an Architecture for an online virtual IT Security Lab," *International Journal of Online Engineering*, vol. 4, May 2008.

[14] G. Vigna, "Teaching Network Security through Live Exercises: Red Team / Blue Team, Capture the Flag, and Treasure Hunt," in *Security Education and Critical Infrastructures*, C. Irvine and H. Armstrong, Eds. New York, NY: Springer US, 2003, vol. 125, pp. 3–18.

[15] T. Yang, K.-B. Yue, M. Liaw, G. Collins, J. Venkatraman, S. Achar, K. Sadasivam, and P. Chen, "Design of a distributed computer security lab," *Journal of Computing Sciences in Colleges*, vol. 20, pp. 332–346, Oct. 2004.

[16] Z. C. Schreuders, T. Shaw, M. Shan-A-Khuda, G. Ravichandran, J. Keighley, and M. Ordean, "Security Scenario Generator (SecGen): A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting {CTF} Events," 2017. [Online]. Available: https://www.usenix.org/conference/ase17/workshop-program/presentation/schreuders

[17] K. Bock, G. Hughey, and D. Levin, "King of the Hill: A Novel Cybersecurity Competition for Teaching Penetration Testing," 2018. [Online]. Available: https://www.usenix.org/conference/ase18/presentation/bock