A Simple Laboratory Environment for Real-World Offensive Security Education

Maxim Timchenko and David Starobinski
Electrical and Computer Engineering Department, Boston University
8 Saint Mary's Street, Boston, MA 02215, USA
maxvt@bu.edu, staro@bu.edu

ABSTRACT

In recent years cybersecurity has gained prominence as a field of expertise and the relevant practical skills are in high demand. To reduce the cost and amount of dedicated hardware required to set up a cybersecurity lab to teach those skills, several virtualization and outsourcing approaches were developed but the resulting setup has often increased in total complexity, hampering adoption. In this paper we present a very simple (and therefore highly scalable) setup that incorporates state-of-the-art industry tools. We also describe a structured set of lab assignments developed for this setup that build one on top of the other to cover the material of a semester-long *Cybersecurity* course taught at Boston University. We explore alternative lab architectures, discuss other existing sets of lab assignments and present some ideas for further improvement.

Categories and Subject Descriptors

K.3.2 [**Computer and Information Science Education**]: Curriculum, Information systems education

General Terms

Security, Design, Experimentation

Keywords

Cybersecurity; Course design; Education; Virtualization

1. INTRODUCTION

The importance of practical, and not only theoretical, education in cybersecurity has long been recognized but has been brought in focus due to events of recent years. Traditionally, the practical skills were developed in expensive hardware-based security labs that allowed experimentation without causing damage beyond the lab limits, but the costs of building and maintainining those laboratories are high.

Recognizing that cost, the academic community has proposed several education-focused solutions that offer hands-on training in virtualized environments, but many of them have drawbacks: some require elaborate setup and centralized server resources [10], or do not provide hands-on experience on offensive tools common in the industry, or have teaching material focusing on advanced, programmer friendly topics that is less suitable for an introductory course [2].

From the industry side, a virtualized environment with a variety of known weaknesses [4] and a full collection of commonly used offensive and defensive tools [3] can be freely downloaded, but the available didactic material using this environment is rather limited to a motley collection of Internet tutorials on specific exploits and does not present a systematic, gradual approach to the field.

In Fall 2013, a semester-length *Cybersecurity* course has been taught at Boston University using a virtualised environment based on freely available and minimally modified virtual machine images. The contributions in this paper, as detailed in the following sections, are: first, a blueprint that allows quick and easy deployment of a simple, secure and isolated environment for a cybersecurity class on commodity workstations or even on students' personal laptops; and second, a coherent collection of lab assignments (topics, text, and full solutions are available) that use standard industry tools and actual known vulnerabilities to offer students an introductory exposure to practice of offensive cybersecurity.

2. COURSE STRUCTURE

The *Cybersecurity* course (EC 521) is taught over one semester (typically 15 weeks) with two scheduled weekly meetings of two hours each, about 60 hours of instruction in total. The goal of the course is to provide a holistic yet technically in-depth examination of security of computers and computer networks, exposing students to adversarial thinking and placing an emphasis on offensive techniques used for penetration testing. The course describes the underlying foundations of popular penetration tools, practical use of these tools, and mitigating solutions against attacks launched with these tools.

About ten distinct topics are covered during the course, with most topics following a three-step teaching pattern:

1. A lecture in a classroom setting introduces the topic and discusses the general methods associated with it. For example, general network structure, desired information, and associated protocols may be covered while discussing the information gathering stage of network security testing.

2. A lecture in a lab setting presents specific cybersecurity tools associated with the topic, with students following along, trying the provided commands and parameters and observing the responses from the tools. For example, `nmap` would be one of the tools presented for performing information gathering.

3. Finally, a lab assignment is given to students to reinforce the material provided in the lectures, to promote further exploration of the topic and the tools with guided questions, and to apply the resulting knowledge to a typical scenario. For example, a lab may ask students to use `nmap` to answer specific questions about a particular host of interest, request them to use a parameter or feature not covered in class, and ask to explain why and how the specific method used achieves its goals.

Students usually have one to two weeks to complete a given lab and work in pairs, promoting discussion and teamwork and reducing the lab grading workload.

The course is intended for first-year graduate and advanced undergraduate students. Its prerequisite is EC 327 (*Introduction to Software Engineering*) and familiarity with at least one programming language and the Linux operating system. Course co-requisites include EC 441 (a first course in computer networking) and familiarity with TCP/IP.

The second and third steps utilise the virtual course environment as presented in the following section.

3. LAB ENVIRONMENT

The laboratory setup used by EC 521 was constructed with the following objectives in mind:

Isolation: the actions of one student should not impact another student's environment.

Persistence: allow the state of a student's environment to be saved between sessions, even if the same lab workstation is used by other students / other courses in the meanwhile.

Hardware independence: it should not matter which lab workstation a given student is using; if a workstation fails, the student's environment should be minimally impacted.

Resource efficiency: minimize resource use (beyond lab workstations themselves).

Security: keep the target (vulnerable) machine off the public Internet; minimize possible effects of student actions on the university intranet.

Reuse: reuse existing components / software / standard industry tools as much as possible.

Simplicity: an introductory course in cybersecurity should not need a complicated environment to achieve its objectives.

The networking laboratory is equipped with approximately 50 workstations with 4 Gb RAM running BU Linux [1], a customized version of CentOS 6. Each machine runs VMWare Workstation 9. Figure 1 shows the connections for a single workstation, which has three virtual machines:

1. The attacker is represented by a *Kali Linux* [3] virtual machine. This distribution is based on Debian Linux, includes a multitude of security audit and penetration tools (including all the tools introduced in the course), and is officially available for download as a VMWare image that boots into a desktop environment. *Kali* also includes a range of GUI tools that make some exploration possible even by users who may not be comfortable with the underlying command line tools.

2. The target is represented by a *Metasploitable 2* [4] virtual machine. This image contains a number of services with known exploits (ftp, http, and so on), insecure settings, and three vulnerable web applications. This VM is also officially available as a VMWare image, and has low memory requirements since it does not offer a graphical user interface.

3. One of the assignments requires a third "zombie" machine. Almost any OS that has networking support would do; we have chosen FreeBSD 6.0 for its low system requirements and for some diversity in the virtual environment (i.e. not 100% Linux). An unofficial VMWare image for this OS is publicly available [7].

Each student was provisioned a copy of the attacker and the target VM on a shared file server. The permissions were set so that a student could access only their own copy of these VMs. These VMs were persistent–changes made by students would be retained throughout the course.

A copy of both attacker and target VMs was also deployed locally on each workstation. These VMs were not persistent and would revert to their original contents whenever

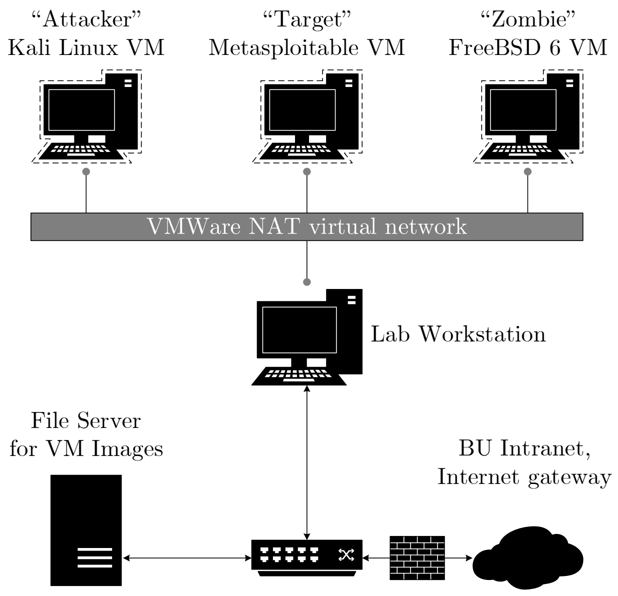

Figure 1: Diagram of the lab environment

[1]`http://www.bu.edu/tech/support/desktop/distribution/bulinux`

they were shut down. The purpose of these VMs was to reduce the load on the network when working with the persistent image was not required, and to let students experiment freely without the risk of damaging their persistent VM. Students were instructed to use either their personal/persistent or the local/temporary VMs as they saw fit.

For the lab requiring a "zombie" VM, the lab instructions provide a sequence of steps to download and run the additional VM on the workstation, offering a hands-on experience of obtaining and exploring a new OS.

The virtual network was configured to use network address translation (NAT), therefore all VMs could access the Internet but external access to VMs was limited (no port forwarding was configured). This was done to let students practice installing additional packages onto VMs and to allow Internet searches, etc. from the attacker VM itself. An even more locked-down approach (that would also reduce the risk of students inadvertently launching an attack on the lab network instead of the virtual LAN) would be to use the host-only networking mode.

4. LAB ASSIGNMENTS

A set of eight lab assignments has been developed for the course [9]. Each lab is accompanied by a comprehensive solution including relevant screenshots and further discussion.

Law and Ethics
: This assignment is closer to a homework than a lab assignment, and asks students to evaluate various actions in context of applicable law and ethics standards of relevant bodies (professional groups, Boston University, etc.)

Introduction
: The first hands-on assignment introduces VMWare (the virtual environment) and the virtual machines used in the course and guides a student through the installation and configuration process, as well as tests understanding of some standard Linux tools and basic networking concepts.

Search Engine Hacking and Social Engineering
: Students are asked to investigate an organization and discover relevant organizational and personal information using a search engine, illustrating the amount of publicly available sensitive information that can routinely be found on the Internet. Another part of the assignment poses several search questions that can be answered using search engine queries with appropriate options, allowing search of historical website content (information that is no longer publicly available but still present in Internet archives) and quick identification of instances of a vulnerable system.

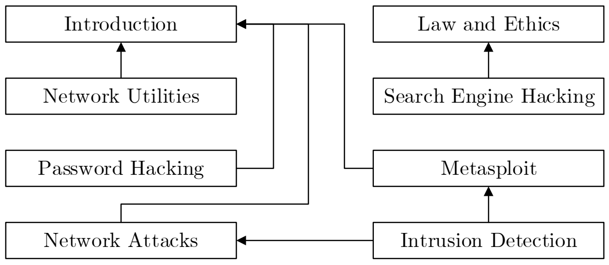

Figure 2: Dependencies between lab assignments

Network Utilities and Scripting
: Students explore various uses of common versatile network utilities (such as `netcat` and `wget`) and do some scripting: in a shell script (utilizing the learned network utilities) and, on a lower level, by interacting with TCP/IP sockets directly from the Python scripting language.

Network Attacks
: This lab focuses on network attack tools included with Kali Linux. Specifically, ways to mount ARP and DNS spoofing attacks on a network are demonstrated. The lab also asks students to do an `nmap` scan uzing the "zombie" technique.

Metasploit
: Using the *Metasploit* framework, students are asked to perform several exploits on Metasploitable's services that are known to be vulnerable, mount different denial-of-service attacks on a web server running on the target VM, and explain why only some of the attacks are successful.

Password Cracking
: The assignment introduces students to an off-line password cracking tool (*John the Ripper*) and an on-line tool (*Hydra*) that can be used to guess passwords to a website requiring authentication. Students are asked to use the on-line tool to find valid credentials to a different web application supplied with Metasploitable, and to crack Metasploitable's password database that they would download after compromising the target machine.

Intrusion Detection
: A popular intrusion detection tool `snort` is introduced, students investigate varous modes of its operation and develop a custom filter that detects a previously studied Metasploit exploit attempt on a specific service.

Whenever possible, the assignments reference topics that were covered by earlier labs. For example, the Network Attacks lab walks students through using `ettercap` to perform ARP poisoning and tests their understanding of the attack. A few weeks later, the Intrusion Detection lab asks students to "perform ARP poisoning" with the expectation they already learned how to do it (or remember the previous lab and go to it for instructions). Figure 2 shows the dependencies among the lab assignments.

5. DISCUSSION AND RELATED WORK

We start the discussion with a brief overview of architectures for cybersecurity lab environments. For a longer survey, see Stewart et al [8].

Originally, cybersecurity labs started from mini-networks containing actual hardware. While offering the most realism, such labs are expensive to acquire and maintain, and tend to fall behind state-of-the-art as new hardware is introduced. The choice of hardware limits the range of possible configurations and caps the number of students that can share a lab without interfering with each other.

The other option is a virtualized environment. Three classes of virtualized environments exist:

- Centralized virtualization: a single server or a group of servers maintain virtual lab environments for all students, who access their environments remotely; one comprehensive centralized virtualization-based system is Tele-Lab [10].
- Cloud virtualization: similar to centralized virtualization, substituting the underlying server resources with data centers of the cloud services provider, and hosted hypervisors with the cloud provider's tools for spinning up, managing, and monitoring virtual server instances. Salah [6] described an instance of this architecture on the Amazon Web Services platform.
- Local virtualization (leveraging the resources of student workstations to maintain their virtual environments; students have local access to their virtualized networks). This is the approach we have adopted.

Both centralized and cloud approaches carry some complexity and cost involved in setting up and maintaining the environment. For organizations already having a server virtualization infrastructure, the former choice could be an attractive one; for organizations with limited on-site resources or technical personnel, the promise of "infrastructure offloading" to a cloud service may appear attractive but it comes with a bill for the resources consumed and does not really alleviate the demand for qualified technical staff to manage the environment.

The method of providing the entire virtual network on a single workstation benefits greatly from advances in capabilities of modern hardware. For example, Stewart et al mentioned being able to run at most three virtual machines using 256 MB each on a host with 1GB RAM in 2009 [8]. At the time this lab was run, the three guest operating systems used 1.5 GB of RAM on a host equipped with 4 GB, using less than 50% of the workstation's capacity.

As RAM and core count of typical lab machines go up while the system requirements of virtualizing a small number of operating systems and network devices remain roughly constant (since the relevant concepts can be demonstrated effectively on older versions of operating systems), hosting such a selection of 10-20 devices (counting user machines and servers, network hardware and filters/firewalls, and peripheral emulators) should be within capacity of a single modern workstation equipped with 16 GB RAM.

As a side effect of using commonly available software in a simple configuration, students could (and some of them did) use their personal laptops running Windows, Linux or OS X (with VMWare Fusion) to run the entire lab environment, including performing most of the lab assignments. There are several benefits to this flexibility: students can experiment with the setup anywhere, not only at the lab; students are working with the host OS that is most familiar to them; and it might well be that their personal laptop is more powerful and therefore runs the lab more smoothly than a lab workstation. If this configuration is allowed, students should be instructed to use host-only networking mode: otherwise, they could inadvertently use dangerous tools on any network they happen to be connected to.

Since we hosted persistent per-student virtual machine images on network storage, we were concerned with the possibility of "long delays in copying large virtual machine images across the network for each student's work" [8]. We did not find this to be a problem in practice. This might be attributable to some students choosing to use local non-persistent VM images in the lab (or their own computers with local images), or to modern virtualization software being able to access only the data it needs from a network hosted disk image, instead of copying the entire image to local storage.

5.1 Scalability

We would like to examine the scalability of each virtualization option (and of the set of labs) in the context of scaling up to a large student audience, such as a MOOC (massive open online course) environment.

Out of the three virtualized approaches, centralized virtualization is the one most likely to run into scaling issues as number of students grows and resource use approaches hardware capacity. Network use between virtual machines can also become a bottleneck, particularly in scenarios that involve large amounts of traffic (DoS exercises, for example). It appears unlikely that most institutions would provision enough spare on-site capacity to deploy the environment for thousands of students.

Cloud virtualization enjoys a flexible and deep pool of resources, but the cost of maintaining the environment scales linearly with the number of students. Particularly in the context of the "open" (free) online course, the cost for the hosting institution can escalate quickly. The complexity of managing a large number of students, particularly when not using any dedicated tools beyond the standard cloud provider's provisioning user interfaces, can also explode in a non-linear fashion. However, it would be possible to develop a specialized interface to handle specific user interface issues and enable both a broad overview and a quick drill-down to a specific student's environment.

In contrast to the above approaches, scaling to a large number of local virtualized environments is fairly easy as each student utilizes one standard computer (either university's or their own) to deploy the full environment. The remaining services (a learning management system hosting the assignments, a file server hosting the VM images) are readily available at MOOC scales. There are examples of MOOCs that have chosen this approach and have run successfully with thousands of students.[2]

One potential issue with a locally hosted virtual environment is that the student's computer must be powerful enough to support it. This will inevitably not be the case for some students, who would have to seek alternative options. An interesting possibility could be to deploy cloud instances only for those students, and have the students cover the costs of running those instances. [3]

5.2 Didactic Materials

There are several documented sets of lab assignments for cybersecurity courses. The set produced by the SEED project [2] is perhaps the largest collection available that is designed

[2]For example, the *Software Defined Networking* MOOC on Coursera has provided a VirtualBox VM containing all the necessary tools for the course.
`https://class.coursera.org/sdn-002`.

[3]A relevant survey from the SDN MOOC:
`https://piazza.com/class/hvjn9udjegm75q?cid=426`

specifically for instruction and uses a unified virtual environment (a version of Ubuntu) for most of the labs. It might be surprising, then, that there is not much overlap between labs developed for this course and SEED labs; only the *Network Attacks* lab corresponds to a SEED *TCP/IP Attack Lab*. Compared to SEED labs, which are often exploratory in nature, our labs offer more guidance on the steps to perform and pose specific questions for students to answer, making them easier to grade and reducing the overall time required to complete a lab.

Another set of labs is the *OWASP Hackademic* [5]. Its original focus is web applications security, and it uses a more involved setup to administer specific challenges. The authors do intend[4] to expand the system to cover virtual machine-based security challenges, but at the time our course was offered the functionality was not available.

There are many papers presenting one or more lab assignments that can be used in a cybersecurity course. For example, Yuan et al [11] introduced two such labs: a wireless network attacks lab and a stack overflow lab. However, with each such work offering only partial coverage of the course's intended topics, it would be a significant effort to reconcile labs from disparate sources (having different setup requirements, writing style, level of difficulty and so on) into a harmonious set for the complete course.

It is worthwhile to note that our course plan includes an *Laws and Ethics* lab early on, before any tools and methods are introduced. Information security skills in general, and offensive security skills in particular, can be used for both good and nefarious purposes, and covering the relevant laws, ethical principles, and guidelines at the outset helps students evaluate their actions. While explicitly not technical, the importance of having an ethics lesson or lab early in a course plan has been pointed out in previous work [1], but such a lesson is absent from both SEED and OWASP lab sets at this time.

5.3 Choice of Virtual Machines

We have chosen to use *Kali Linux* and *Metasploitable* because these readily available images represent and teach the set of tools commonly used in the security industry and typical weaknesses that can be attacked using those tools respectively. Since both VMs were designed from the outset to be usable by individuals learning at their own pace and on their own equipment, the recommended environment already had to exclude anything that requires elaborate configuration or a complicated process.

Choosing a stable and popular environment that had all the necessary tools and required minimal modification enabled us to focus on creating content for the course instead of dealing with setting up a more elaborate system.

5.4 Choice of Virtualization Software

We used VMWare Workstation because the virtual machine images for both *Kali Linux* and *Metasploitable* were only "officially" available in VMWare format. VMWare Player can be used for free by students to run the images on their computers, and a compatible Mac version (VMWare Fusion) exists. Furthermore, the university already had a license for VMWare software, so licensing cost was not a concern.

Image compatibility of virtualization software is constantly improving. We successfully converted the VM images into VirtualBox `.ovf` format and verified several lab questions to work. The instructions to do so, and the results of our testing in Parallels on a Mac, will be posted to GitHub [9].

6. STUDENT FEEDBACK

Towards the end of the semester, students were asked to fill an anonymous survey evaluating several teaching-related and course-related questions. Each question uses a scale from 1 (poor) to 5 (excellent). Out of 38 students, 27 have chosen to fill the survey; of those, 18% identified as undergraduate and 67% as graduate (15% did not answer the question). Figure 3 shows the responses to several of the questions asked on the survey.

The survey also allows students to write a freeform feedback about the course. The following are a few chosen responses:

> "The course was fun. I enjoyed the labs a lot. I definitely learned quite a bit about cybersecurity from these labs alone."

> "Strengths: in-class exercises and labs. Practical applications of the course tools helped to clarify somewhat semantic [sic] lecture material..."

> "Strong: good organization and explanation. Fun labs. Weak: even more hands-on would be great."

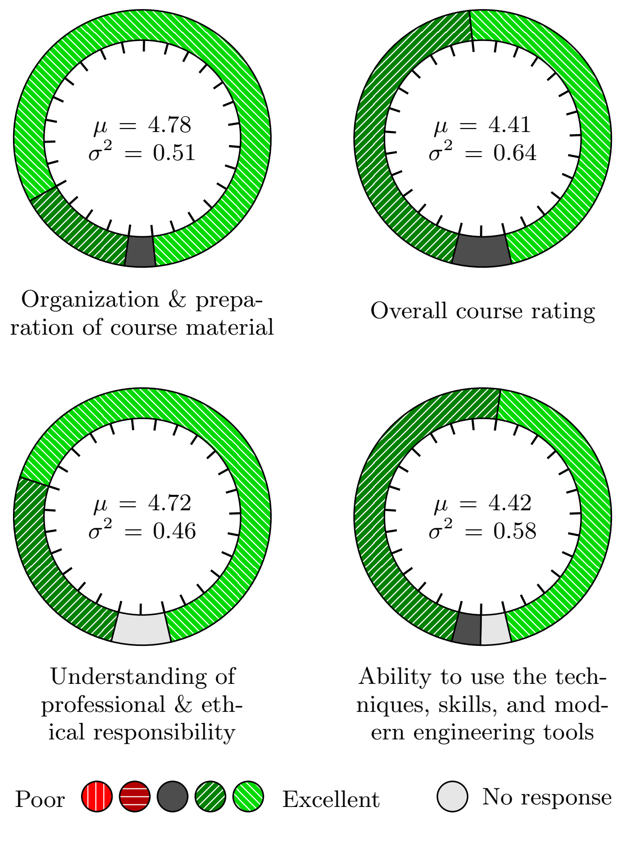

Figure 3: Student survey results (27 respondents)

[4]"We intend to expand the available challenges with additional scenarios that involve cryptography, and even vulnerable systems implemented in download-able virtual machines." - `https://www.owasp.org/index.php/OWASP_Hackademic_Challenges_Project` as of March 10, 2014

A few students mentioned that the labs were too easy. Due to diverse background of students, we consciously chose exercises and provided guidance such that our labs would be accessible to most students, accepting the fact that advanced students would find some of the labs too easy. A related survey question shows that a larger number of students do find the labs challenging and, on average, the difficulty is reasonable: 14 students (52% of respondents) rated the level of difficulty as "average", 8 students (30%) rated it "too difficult" and only one student rated the course "easy".

7. CHALLENGES AND FUTURE WORK

While Kali Linux (the attacker VM) has been updated every few months so far, the Metasploitable VM has been released in mid-2012 and saw no updates since. It remains usable as a snapshot, but if any changes have to be made to the image (for example, adding a package that was not installed by default) some workarounds and configuration changes are already needed, as the base OS used for Metasploitable has transitioned from supported to archived status in 2013. As such, and without explicit commitment to supporting Metasploitable going forward, its future and expansion are uncertain.

We have spent some effort modifying a Metasploitable image in an attempt to create a different environment for the course's final exam (different available services, exploits, passwords and so on), but adding new exploits was not possible due to limited time. The task would have been easier if the VM was designed in a modular, customizable manner–for example, if it was possible to install vulnerable services from a pre-built package repository.

A single Linux VM is enough for a first cybersecurity course since it can demonstrate the relevant concepts. However, the current computing environment is characterized by a multitude of popular architectures and operating systems. A more realistic environment would include other vulnerable virtual machines based, for example, on Microsoft Windows XP and Mac OS X; and different architectures, for example Android on ARM providing an example of a smartphone platform or Linux-based OpenWRT on MIPS32 for an example of a home router platform. We are not aware of an effort to provide such images specifically for cybersecurity training (with known security holes and supporting didactic material), even as vulnerabilities in smartphone operating systems and embedded devices are making news throughout the tech industry.

Specifically for Windows XP, Metasploit exploits are available for the operating system itself (unpatched or on earlier service pack level), but obtaining known vulnerable (old) versions of third-party software has proven more difficult than authors expected. Using a proprietary OS also carries licensing issues, but many institutions would have Windows XP licenses available from previous deployments.

From the scalability perspective, the biggest challenge in scaling a cybersecurity course to a massively online audience is likely to be in grading. MOOC classes typically use automated grading; developing a method to collect relevant output from different machines of the local virtualized environment, automatically grading, and providing meaningful feedback for each of the proposed subject areas would be a big step forward towards enabling a MOOC instance of the course.

Acknowledgements

The authors would like to acknowledge the contributions of James K. Goebel, including the initial idea of switching to the virtual machine-based environment and technical aspects of the implementation, which are not described in this paper since they are likely to be specific to Boston University's IT services configuration and therefore are of less general interest.

Yuguang Li and James Christianson wrote the initial versions of some of the lab assignments mentioned in this paper and helped in testing the virtual course environment.

This research was supported in part by NSF under grants CNS-1012910 and CNS-1409053.

References

[1] Thomas Cook, Gregory Conti, and David Raymond. "When good Ninjas turn bad: Preventing your students from becoming the threat". In: *Proc. 16th Colloquium for Information System Security Education.* 2012, pp. 61–67.

[2] Wenliang Du. "SEED: hands-on lab exercises for computer security education". In: *Security & Privacy, IEEE* 9.5 (2011), pp. 70–73.

[3] Offensive Security Ltd. *Kali Linux.* 2014. URL: `http://www.kali.org` (visited on 03/05/2014).

[4] HD Moore. *Metasploitable 2 Exploitability Guide.* 2012. URL: `https://community.rapid7.com/docs/DOC-1875` (visited on 03/05/2014).

[5] Alexandros Papanikolaou et al. "A Hacker's Perspective on Educating Future Security Experts". In: *Informatics (PCI), 2011 15th Panhellenic Conference on.* IEEE. 2011, pp. 68–72.

[6] Khaled Salah. "Harnessing the cloud for teaching cybersecurity". In: *Proceedings of the 45th ACM technical symposium on Computer science education.* ACM. 2014, pp. 529–534.

[7] *Some VMWare Images.* 2013. URL: `http://www.thoughtpolice.co.uk/vmware/` (visited on 03/05/2014).

[8] Kyle E. Stewart, Jeffrey W. Humphries, and Todd R. Andel. "Developing a virtualization platform for courses in networking, systems administration and cyber security education". In: *SpringSim '09: Proceedings of the 2009 Spring Simulation Multiconference.* San Diego, California: Society for Computer Simulation International, 2009, pp. 1–7.

[9] Maxim Timchenko and David Starobinski. *EC 521 Cybersecurity Lab Assignments.* 2014. URL: `http://github.com/maxvt/cyberlabs` (visited on 11/20/2014).

[10] Christian Willems and Christoph Meinel. "Tele-Lab IT-Security: An Architecture for an Online Virtual IT Security Lab". In: *International Journal of Online Engineering* 4.2 (2008).

[11] Xiaohong Yuan et al. "Developing Faculty Expertise in Information Assurance through Case Studies and Hands-On Experiences". In: *System Sciences (HICSS), 2014 47th Hawaii International Conference on.* IEEE. 2014, pp. 4938–4945.