Proceedings of the 16th Colloquium for Information Systems Security Education
Lake Buena Vista, Florida June 11 - 13, 2012

Teaching Network Security Using VITAL

Efstratios Gavas[1] and Keith O'Brien[2]

[1]*Polytechnic Institute of NYU*
[2]*Cisco Systems*

***Abstract*—Network Security is a complicated course to teach requiring extensive hands-on experience to fully develop the students knowledge base. To help facilitate comprehensive lab exercises, NYU-Poly developed VITAL – a Xen-based, remotely accessible, open source virtualization platform designed around the classroom environment.**

I. Overview

At Polytechnic Institute of NYU, a graduate level *Network Security* class is offered as part of the Master's degree program in Cybersecurity. While it is a core course for this degree, it is also routinely taken by those pursing a general Master's degree in both Electrical Engineering and Computer Science. The students enrollment is typically a mix of full time students, who have very little real world experience in network security, along with seasoned professionals who are pursuing the degree on a part-time basis.

Seeing a need for hands-on learning, we developed the Virtual InformaTion Assurance Laboratory (VITAL) Virtual Lab, a web-based virtualization environment built for teaching. VITAL was started because of the need for a simple and easy to use virtualization environment designed for the classroom setting. The authors of this paper have been involved in both the development the VITAL system as well as the network security curriculum. Although other virtualization environments exist, these environments have been designed around the operational needs of a server room such as VMWare [4] and Microsoft Hyper-V [3], or research centered such as DETER[1] and GENI[2]. In contrast, VITAL was designed to be simple to use for both students and professors without an extensive IT support staff. Additionally, because VITAL is built on top of the Xen virtualization environment, we are able to take advantage of both the operational features and the open source nature of the system.

II. Challenges of Teaching Network Security

A. Student Background

The highly varied student background presents challenges in striking a balance between those who are completely new to the security field and those that already have some experience. We took the middle road and presented topics in which some students are going to have to dedicate time outside of the normal budgeted time for out of class work. Linux acumen, as an example, is a heavy requirement in order to complete the lab portion of the class. We spend very little time in bringing those who do not have a Linux background up to speed. The expectation is set at the onset that the material is going to have to be learned on their own. Further, it is enforced that anyone who plans on a successful career in the network security field is going to need the initiative to consistently spend their own time on education even after their formal education ends.

B. Teaching Offensive Attacks

Although various pedagogical differences exist regarding the best method(s) for teaching cyber security[8], [5], [10], we take the approach of actively teaching offensive attack techniques. We believe the most effective way for a student to properly learn defensive network design concepts without the understanding of how networks are attacked. In other words, "know your enemy." Teaching hacking techniques can present challenges with school administration who might not have a understanding of the importance of establishing this baseline in attack techniques and certainly has the potential to be controversial. It does need to be enforced that ethics and responsibly are expected from the students and that all techniques taught are to be confined to a properly segmented network attack laboratory. Even experimenting with techniques "at home on their own network" needs to be actively discouraged. Finally we remind students of the academic and legal consequences of not adhering to ethics in the class.

III. Importance of Hands-On Learning

The Network Security class was designed to have a very heavy emphasis on laboratory work. A typical semester consists of at least 8 labs which require 10 hours of lab work each. In setting pedagogical goals for the class, it was found that many students who have taken the class never experienced first hand how networks and computers are attacked and the techniques used. The student are given access to a *standard network* as illustrated in Fig. 1 with full administrative access to six virtual machines on the VITAL.

With that in mind, the lab work was designed such that the first 4 labs take the students through the *Network Attack Methodology*. While there are many variations of this framework we present it as:

1) Recon / Information Gather
2) Scanning / Enumeration

ISBN 1-933510-95-1/$15.00

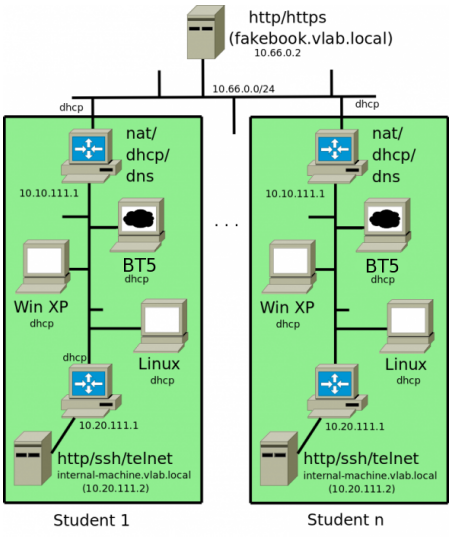

Fig. 1. Standard Student Network

3) Vulnerability Identification
4) Exploitation
5) Post Exploitation / Keeping Access
6) Covering Tracks

Following this framework the first lab has the students perform reconnaissance on a network containing a few Linux and Windows XP machines. Since this course is not intended to cover advance exploitation techniques, using Windows XP sets the appropriate level of difficulty in the exploitation step as one part of teaching the attack cycle. The lab requirements are not prescriptive and instead instruct the student to *think like an attacker*. The first labs objective is to find all of the machines which exist on the network and along with services listening on those machines. Using a VM of Backtrack 5 as the attack platform, the students use tools such as nmap and Nessus to perform network recon. The lab is completed when the student has a list of potential vulnerabilities on the target Windows XP machine. At this point the students have first hand experience with Steps 1 through 3 of the Attack Methodology.

We pickup in the next lab with Step 4 – *Exploitation*. The instructions given to the students are to achieve remote shell access on the Windows XP machine, transfer a file to that machine from the attack platform and finally take a remote screen shot of the desktop. While most students will use pre-built exploits within the *Metasploit* framework; we do not constrain them. They receive credit when the objective is met. If they want to write their own exploit or use a targeted tool outside of *Metasploit* that is certainly kept open as an option. The only rule is to exploit the target given the resources provided in our lab environment.

We complete the Attack Methodology in the final lab of the series by using the remote shell connection to transfer a root kit to the target. We use *HackDefender* as the example rootkit. The students are instructed to prove that they are maintaining access to the XP machine even in the presence of target reboots. Further, they are instructed to hide the rootkit processes from casual exploration on the client side.

It is important to note that the goal of this portion of the course is not to educate the students on attack tools. The pedagogical goal is for them to acquire first hand experience in executing a complete network attack which follows the Network Attack Methodology presented in class.

The remaining labs are more focused on the defensive technology which can be employed to prevent these offensive attack techniques. Even though, we still maintain a offensive perspective to all lab exercises. Using the example of IPsec and SSL which is covered in class, instead of having the student simply setup VPNs, we have them execute an attack on SSL. The goal is two fold: 1) By attacking the protocol we reinforce their understanding of how the protocol works. 2) We further reinforce that no technology in the security space is foolproof. Many students enter the class with the incorrect notion that SSL in the browser protects them from network eavesdropping. The lab has the students perform a *man-in-the-middle* attack on a web client using *arp spoofing*. Once they are in the middle of the traffic we have the student use a tool called *SSL Strip* which removes SSL encryption without the end user noticing and without any confirmation pop-ups. We further bring this into their world by using a lab confined site called "Fakebook" in which they target their attacks.

Course feedback from students has been overwhelmingly positive. The labs have opened student interest to the field of network security. We have many examples of students previously taking the class as an elective completely switching their academic and/or professional path to that of Cyber Security after completing the course. Grabbing the students interest in the subject with real world lab examples has been proven to be the key in obtaining this result.

A. Impact of VITAL

VITAL has provided hands-on tools and capabilities for an increased number of Cyber Security courses and activities at NYU-Poly. Following the Fall 2011 Semester, we conducted a survey of students to measure the impact of VITAL. Of the 45 responses to the survey, a majority the students regarded the use of VITAL as beneficial.

In Fig. 2 one can see over 80% of the responding students *strongly agreed*, or *agreed*, that VITAL had a positive impact on their learning experience. Additionally, almost 60% of the responding students *strongly agreed*, or *agreed*, that the web-based interface was better than a local VM solution. Further, over 90% of the responding students felt VITAL provided them good hands-on experience.

In Fig. 3 VITAL also showed equally strong responses to specific course objectives. 65% of the responding students *strongly agreed*, or *agreed*, that VITAL was effective in helping them learn *TCP/IP and Network Programming*. Further, 80% to 90% of the responding students *strongly agreed*, or *agreed*, that VITAL was effective in helping them with the other technical class objectives.

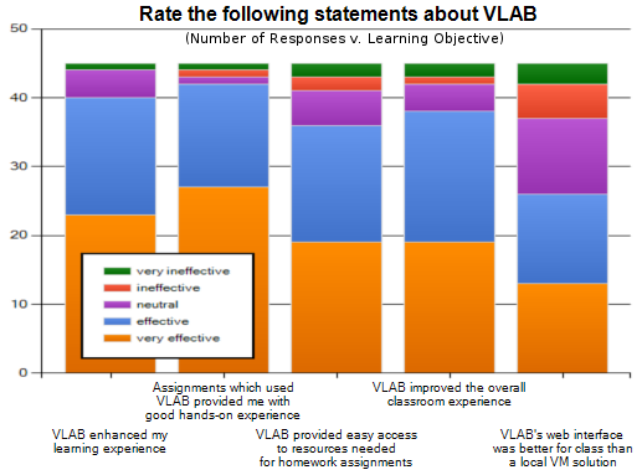

Fig. 2. Impact of VITAL on Learning

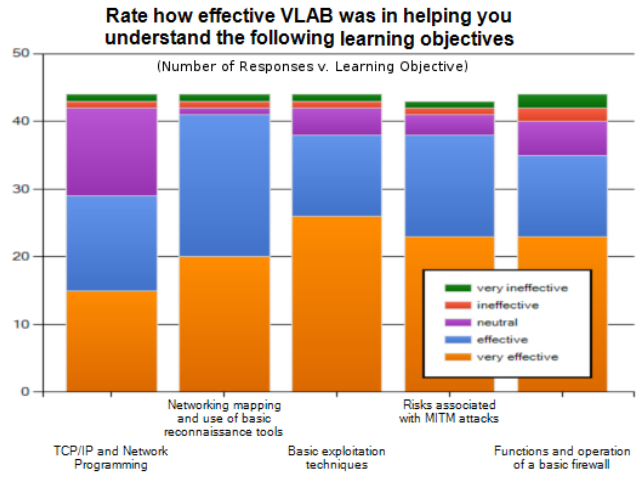

Fig. 3. Effectiveness of Teaching Learning Objectives

IV. Benefits of Opensource Virtualization

VITAL development would not have been possible without the flexibility of open source software. We chose Xen over other virtualization platforms because of the flexibility provided by the open source nature of the product. Since the VITAL project began, other virtualization platforms have developed to address the needs of the classroom. However, at the start of developing VITAL no other platforms tailored for the classroom existed; we had the flexibility to start our project because Xen is open source. Additionally, since the platform has since obtained commercial backing through Citrix XenServer, we continue to believe Xen is the most appropriate virtualization platform for the classroom environment.

A. Motivation

The VITAL platform was extended from earlier work[9] done at NYU-Poly to teach network security in order to allow instructors to set up appropriate laboratories based on virtualization in a simple and efficient manner. Instructors are then able to use these laboratories to provide hands-on experiences to their students in cyber security related courses. The system is globally accessible, easy to maintain and use, robust, extendible, and promotes the dissemination of cyber security understanding, sharing of lab assignments and experiences between diverse institutions.

Building laboratories for security education can be challenging as typical projects require a student to have complete access to an entire network of computers. For instance, a firewall configuration project requires a student to be given access to a network that consists of one or more hosts that model computers on the Internet, one or more computers that model the hosts on the internal network protected by the firewall, and the firewall itself. To complete the assignment, students will first need to configure the firewall with appropriate rules. Following this step, the student needs to run tests to determine if the firewall permits access to authorized services, while denying access to others. To complete these steps, the student will need administrative access on the firewall itself, as well as the internal hosts that run services accessible from outside. She may also need administrative access to run testing tools on the "outside" hosts. Network monitoring, network forensics and intrusion detection are other obvious examples of projects that require a similar level of access for all students in the course.

There are several challenges in providing the type of access described above. First, students may make configuration errors that have the potential to damage the entire operating system, and hence render a computer unusable. Thus, it is necessary to rely on techniques that can restore damaged operating system installations without any significant effort on the part of the course staff [1].

A second challenge is that some students may misuse administrative privileges to carry out activities that harm other students in the class (or others on the Internet) by snooping on their traffic or carrying out attacks. To eliminate this threat, the networks used by each student should be disjointed and isolated. However, the cost of developing such physically isolated networks can be prohibitive for moderate to large classes. For instance, to support a class of 50 students, the above experiments will require in the neighborhood of 300 computers. Thus, the hardware costs alone could amount to hundreds of thousands of dollars.

The third challenge is one of cost, which includes acquisition as well as operational costs. Most institutions lack the resources to purchase the necessary hardware and software to set up such a laboratory, since the above requirements imply that the lab should be large in comparison with the number of students, e.g., supporting many computers and dedicated networks for each student. Even worse, the operational and administration costs, which typically dwarf acquisition costs by a factor of 5 to 10, cannot be afforded even by some of the biggest institutions.

[1] we use the term "course staff" to refer to instructors, teaching assistants or dedicated system staff that are involved in setting up and administering the laboratory and assignments for a course.

B. VITAL Goals

The pedagogic value of interactivity is well established[6][7]. Historically, many types of systems started off in the batch processing paradigm and then transitioned to interactivity. As a system grows in popularity, widespread acceptance requires the transition to interactivity. During any sort of experimentation, a user wants feedback as soon as possible to minimize time spent on frustration. An interactive environment provides the tight feedback loop needed for students.

The overarching goals of the VITAL project are:

- Facilitate the adoption and use of the virtual lab courseware by different institutions.
- Make it easy for faculty from different institutions to add hands-on components to their courses by using our laboratory infrastructure.
- Enrich student experience in cyber security, and endow them with the skills they need in the marketplace.
- Attract more students to cyber security, and improve retention of existing students.

In order to achieve our goals we developed an extendible software platform for a virtual cyber security laboratory. The software platform has the following properties:

- Easy to install, maintain and can be used with minimal hardware.
- Open source allowing users to extend its functionality.
- Bundled with a set of prepackaged assignments, virtual machine disk images and lab configuration templates.
- Allows a community of users to contribute assignments which can be downloaded and used by others with minimum effort.

Without these properties, the effectiveness of the project for educational purposes will be minimized.

C. Flexibility of Tools

Since Xen is Linux based, we are able to take advantage of standard Linux tools for development, networking and monitoring. This provided us with significant flexibility to solve a number of problems. Operationally, we has access to the whole suite of monitoring tools available under Linux such as *Munin*, *iftop*, *tcpdump* and others to troubleshoot problems which occur. This flexibility is also available with regard to development tools that are available. Not only can we rebuild the Xen kernel for our systems, but we are also able to use all the userland development tools such as *PHP*, and *Bash scripts* for operational tasks. Further, we additionally have access to the rich set of Linux networking tools which makes the design, creation and monitoring of student networks easy and robust. All of this means that VITAL is a cost effective system to operate which does not require overly complex or specialized skills to maintain.

V. VITAL Architecture

A. Overview

VITAL is composed of three (3) main components which allow it to virtualize a wide range of operating systems and networks configurations:

- *Controller* – Handles the processing of all management operations for the VMs (eg. start, stop, restore).
- *Gateway* – Provides users web access to VM consoles, using VNC, while isolating access from the internet.
- *Xen Nodes* – Cluster of servers used to provide virtualization service.

The VITAL system utilizes a PostgreSQL database to store configurations and state information for the VMs and courses, as well as an *Network File System* for sharing of VM disk images across all Xen nodes in the cluster.

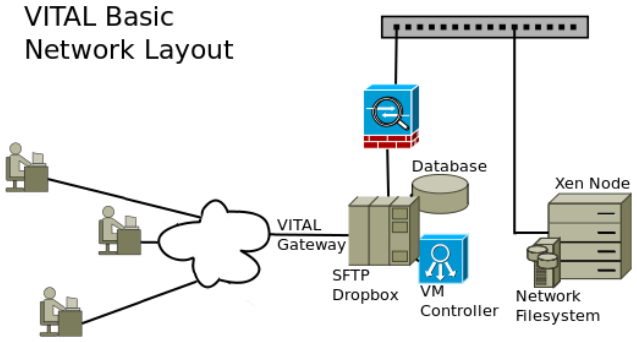

Fig. 4. Basic VITAL Configuration

The system has been designed to be very scalable, and support three general configurations: *minimal*, *basic*, and *full*. In the *minimal configuration* all of the components are configured on a single server. This configuration presents the lowest hardware requirements, but requires careful attention to network resources and VM access to ensure proper isolation from external networks.

For this reason, the *basic configuration*, illustrated in Fig. 4, is the recommended setup for resource constrained installations. Since the key hardware difference between the *minimal* and *basic* configuration is the separate *Gateway* machine, which does not require a modern processor with support for hardware virtualization, nor significant memory and disk requirements, an older (possibly spare) machine can be used to minimize the effective cost of the *basic configuration*. Further, the *basic* and *full* configurations are largely similar with the exception of the additional Xen nodes. As a result, if more capacity is needed, a transition to the *full* configuration is relatively straightforward.

In the *full configuration*, illustrated in Fig. 5, the clustering of *Xen Nodes* and expanded *Network File System* allow for an increased number of VM which can be run concurrently. The *Gateway* server does not depend on the CPU, memory, or network utilization of the VMs, but instead mostly depends on the relatively minor VNC resources required to access each console. Therefore, the *full configuration* can be scaled-up easily by adding more *Xen Nodes* or expanding the capacity of the existing nodes without major impact to the overall system performance.

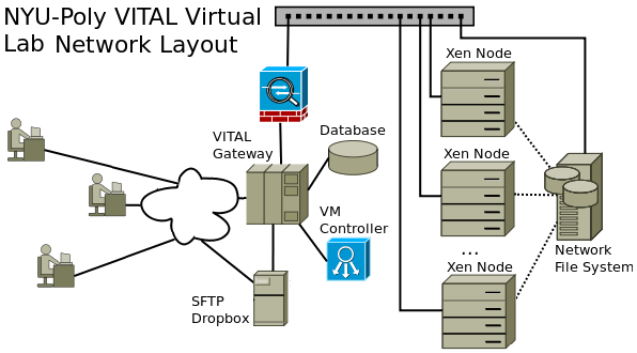

Fig. 5. Full VITAL Configuration

B. VITAL Code Release

All of the source code for the VITAL environment has been released under *GNU GPL version 2* and NYU-Poly will be maintaining a public website for all deployment material. Lab materials (ISOs, VM images), and lessons are available for download along with the installation materials and documentation[2].

A key focus in our effort has been to build a free, open source, and simple platform to facilitate the teaching of cyber security. Along with the development of the platform, we have worked with our partner universities, and others, to build a collaborative community where lessons and tools are shared. By encouraging collaboration, we can leverage our work, and the work of others to build a complete lesson plan along with a safe learning environment.

To help facilitate the development of the community, NYU-Poly has setup a public wiki[3] to maintain all lessons and course materials. The wiki allows open registration and public access to all materials in order to promote wide adoption in cybersecurity curriculum. Other instructors and universities are encouraged to add content and contribute lessons. To seed the community activity we have added six (6) lessons used in our *Network Security* course to the wiki along with the corresponding VM images.

C. Future Work

In addition to continuing to expand the number and scope of lessons available on the wiki, VITAL code development with continue to enhance features and usability from the student, professor, and administrators perspective. The main work planned from the students' point of view is to change the way the VNC connection is handled by the browser from the current Java Applet solution to HTML5 using something like *Guacamole*[4]. The planned additions from the professor's point of view center around better reconfiguration and monitoring. Lastly, we would like to migrate the backend Xen nodes from version 3.4.2 to the newer 4.0 branch for better stability and alignment with main development efforts.

[2]https://vital.poly.edu/release/
[3]https://vital.poly.edu/release/wiki.html
[4]http://guac-dev.org/

VI. Conclusions

The use of VITAL to teach Network security has been largely a success. We have been able to allow the students to develop hands-on experience with the six phases of the *Network Attack Methodology* in a safe and effective manner without the need for elaborate or expensive configuration on the student's side. Additionally, VITAL has allowed for much greater flexibility and range of assignments than would have been possible from local virtualization options and simpler remote classroom solutions than are available from other solutions.

Operationally, one lesson learned was that regardless of how many times the students are told not to wait until the last day or two before the lab is due to avoid server load, over half of the class will wait. Although this many not be surprising information to hear for educators, this is important from the hardware provisioning perspective. When designing the system, peak load should be considered around 50% of the allotted virtual machines assigned. Further, we found that students often leave there VMs running without consideration for the resources which remain allocated to them. A technique which we implement with some success to reduce this problem was to create a *cron script* to auto-shutdown VM running for more than a certain amount of time. During assignments which have particularly heavy resources usage, we have found the need to make the time window as short as 3 hours.

Another important lesson learned was to allow students to self-administer their network when possible. Offloading administrative tasks, such as VM reimaging, to the students saves both student and administrator time. VITAL does this while keeping in mind our objective of a simplified user experience. In the case of *VM reimaging*, this is done with a single button from the user interface without any complex configuration or setup parameters.

Finally, we would like to reiterate our belief that hands-on labs are critical in teaching network security. Without this experience the students do not have a full picture of an attacker's mindset and capabilities. Our surveys further indicate that web-based virutalization solutions are the preferred method for use by the student and provides us the most flexible options in terms of student network deployment.

VII. Acknowledgements

We would like to thank the National Science Foundation for their gracious support and funding of this project though capacity building grants, and Vikram Padman for his initial work building the system as part of his Masters Thesis. Additionally, we would like to thank Luis Garcia and Evan Jensen for their administrative and development efforts, as well as Karthikeyan Sekar for his help keeping the system running.

REFERENCES

[1] Deterlab (DETER). http://deter-project.org/.

[2] Global environment for network innovations (GENI). http://www.geni.net/.

[3] Microsoft Hyper-V. http://www.microsoft.com/en-us/server-cloud/windows-server/hyper-v.aspx.

[4] VMWare products. http://www.vmware.com/products/.

[5] S. Fulton and D. Schweitzer. A concept focused security lab environment. In *Proceedings of the 15th Colloquium for Information Systems Security Education*, CISSE 2011, 2011.

[6] C. E. Irvine. Amplifying security education in the laboratory. In *Proceedings IFIP TC11 WC 11.8 First World Conference on Information Security Education*, pages 139–146, 1999.

[7] P. Mateti. A laboratory-based course on internet security. In *Proceedings of ACM SIGCSE Technical Symposium on Computer Science Education*, 2003.

[8] D. P. Nanette S. Poulios. Scenario based exercises in ia courses. In *Proceedings of the 15th Colloquium for Information Systems Security Education*, CISSE 2011, 2011.

[9] V. Padman, N. Memon, P. Frankl, and G. Naumovich. Design and implementation of an information security laboratory. volume 2, 2003.

[10] B. A. Pashel. Teaching students to hack: ethical implications in teaching students to hack at the university level. In *Proceedings of the 3rd annual conference on Information security curriculum development*, InfoSecCD '06, pages 197–200, New York, NY, USA, 2006. ACM.

ISBN 1-933510-95-1/$15.00