AC 2012-4887: WORK-IN-PROGRESS: CREATING AN INTRUSION DETECTION EXPERIMENTAL ENVIRONMENT USING CLOUD-BASED VIRTUALIZATION TECHNOLOGY

Mr. John M. Jones, East Carolina University

John Jones is currently an Instructional Technology Consultant with the Department of Technology and Computer Science at East Carolina University. He has worked in the IT industry for 18 years in varied roles such as software design, IT manager, security, infrastructure management, systems administration, webmaster, and part-time faculty.

Dr. Te-Shun Chou, East Carolina University

Work In Progress: Creating an Intrusion Detection Experimental Environment Using Cloud-Based Virtualization Technology

Abstract

This paper discusses the latest capabilities of lab automation software and how such software could enhance the learning environment for an Intrusion Detection class. A lab automation system with virtual networking would enable a more realistic environment akin to the real world. Our design utilizes VMware software such as ESXi server, Virtual Center and Lab Manager to provide a robust virtual environment. The experimental results suggest an improved classroom environment for learning Intrusion Detection Systems and related software; in addition, the classroom environment requires much less setup time on the part of the student.

Keywords: virtualization; intrusion detection; cloud computing

1. Introduction

An age-old concern for every educational institution is how to provide the student with the best environment for learning. For teaching network security related courses, the network environment could be built using a collection of physical equipment such as servers, hubs, switches, bridges, routers, and intrusion detection and prevention system (IDPS) sensors[1]. This approach provides students with an actual network to carry out experiments; however, the equipment is expensive and it is time consuming to physically set up all of the network devices. Instead of using physical equipment, virtualization technology is employed to build a network with multiple virtual machines[2,3]. Within a single physical host machine, multiple virtual machines are created and operated simultaneously. In each virtual machine, applications and services are implemented and the virtual machine executes the code just as a physical machine would. This approach eases the load of network administration as mistakes can be easily fixed while the network stays up and running. When a network change is required to conduct desired cyber attack experiments, it can be easily reconfigured in a virtual environment.

The focus of this research is to improve the academic environment for intrusion detection courses by utilizing powerful virtualized environments (VEs). Attention will be focused on the concept of utilizing web based lab automated VEs to reduce time spent by students on setting up environments and how VEs can provide a more cogent learning environment. In this research a VE will be set up so that a student may utilize it from day one with little effort. The environment emulates a typical physical network that includes one student environment and one instructor environment. In the instructor environment, the instructor could launch attacks to the student's environment so the student can learn how to identify unexpected attacks. Students are required to configure a firewall or IDPS to mitigate the attacks. As a result, students can gain direct knowledge of intrusion detection and incident response. This research will not only provide

instructors with information to deliver capable educational environments for teaching future IT security professionals but also provides students with a more immersive educational experience which in turn will better prepare them for the real world. This paper is organized as follows. Section 2 describes an experimental methodology where a proposed virtual computing environment is recommended over physical or traditional virtual machines. Section 3 lays out the proposed virtual environment for both student and instructor.

2. Experimental Methodology

Lately, the technology industry is seemingly immersed in discussion over cloud computing. It is not always easy to define cloud computing, but for the purposes of this paper cloud computing means that the computational power required for the proposed virtual environment is provided somewhere other than the instructor or student computer. Depending on the types of services provided, the cloud models are generally classified into three categories. They are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS offers users ready-to-use applications. PaaS typically provides users with a full operating system. IaaS allows users to use computing infrastructure such as servers, software, and network equipment. The cloud model that we utilize is IaaS, which provides users a set of networked virtual machines for carrying out intrusion detection experiments.

A cloud based virtual environment would require a far lower threshold to setup and reduce computing requirements on the student's equipment verses a physical computing environment or traditional virtual environment. As Casado points out, giving students a hands-on experience normally requires deploying and debugging parts of a network infrastructure, such as a router or a firewall[4]. A risk to privacy and security is possible with the level of access required for students to be able to log and generate raw network packets; such activities would normally require a student to have a dedicated computer with ability to modify the system at the kernel level. The author implies that complexities can build quickly on the student's end; the proposed system in this paper offers an alternative. In a traditional virtual environment, computing requirements can quickly escalate. Some modern system requirements are listed in Table 1.

As can be seen in Table 1, utilizing just four operating systems could require as much as 3.7 GHz of CPU, 3.25 GB of RAM and 40 GB of hard drive space, in addition to the reqirements of the host operating system; this is beyond the means of the average student system. Though computer capacity grows over time, so do the operating system and application software requirements.

2.1. Proposed Architecture

The proposed system architecture listed in Figure 1 is one of many ways an IDPS can be implemented within a virtualized environment. The system includes one student environment and one instructor environment and they act as attacker and defender, respectively.

The student environment includes two Windows platforms, two Linux platforms, a firewall appliance and a Network Intrusion Detection System (NIDS) appliance. The specific version of each component is intentionally left out because each platform can be easily updated to keep up with current industry trends. The instructor's environment includes two Operating Systems either of which is able to send normal and attack traffic to the student's environment, something that is

Table 1. Operating System Minimum Requirements

Operating System	CPU	RAM	HD
Windows Vista	1 GHz	1 GB	15 GB
Windows 7	1 GHz	1 GB	18 GB
CentOS Linux	1 GHz	1 GB	4 GB
Ubuntu Linux	700 MHz	256 MB	3 GB
Total	3.7 GHz	3.25 GB	40 GB

not practical and nearly unfeasible in a traditional configuation. The lack of practicality comes in to play when attempting to provide the sort of network environment required for the proposed architecture; an architecture where a professor could launch attacks in to the student environment.

In a traditional environment, launching attacks in to a student environment would require deliberate, lengthy and likely custom configuration for each student's environment; if the setup were not configured correctly, the professor's attacks could have an effect on the students' personal system, something that should be avoided at all costs. The lack of feasiblity was eluded to in Section 2.1, the computational power requirements of the proposed system are simply beyond what most desktop environments were designed for. Such an architecture could only be practical and feasible with a virtual lab environment or the forthcoming "cloud computing" environments.

2.2. Lab Automation Software

As of this writing, there are three commercial lab management software packages available, they are VMware Lab Manager, VMLogix Lab Manager, and Surgient Lab Management platform. All of the companies listed offer an academic discount for the software but VMware allows the use of Lab Manager for free under the VMware Academic Program (VMAP). In fact, all the software necessary for our research was provided for free.

Lab Management software was seen as a building block to the latest cloud offerings. Lab management and virtualization software was important enough that all the companies listed above have been aquired by larger companies.

Two out of three packages are soon to be changed to incorporate cloud computing in the name. VMware corporation has moved many employees from the development team of Lab Manager to the new Cloud Director software team and is incorporating lab management features in to Cloud Director. Cloud Director is expected to eventually replace Lab Manager. We anticipate many cloud management software packages will be capable of supporting lab automation features in the future.

Lab automation features can help with the following issues: multiple operating systems (OSes) in a single enviroment, complex networking between OSes, pre-configured vulnerable targets, installed base of software, and insufficient computing power. Multiple operating systems can be provided to each student in cases where a special environment is needed. Complex networking

can be assigned between each operating system provided to a student or instructor. A student could be pre-set up with vulnerable targets that only the assigned student is authorized to use. The student environment could also be set up with a pre-installed base of software to utilize. Computing power would be provided on the organization's end so as to avoid burdening the student's personal computing resources.

The first asset of lab automation software, having multiple operating systems in an environment with networking between all elements, is easily set up. The vulnerable targets can be kept in a "fenced" environment safe from other environments and even from other students in the same class. Fencing is a means of walling off an environment to protect from data going in or out of the network unless specifically planned.

Virtual machines within a fenced configuration would be assigned preconfigured internal IP addresses. Each VM would also be assigned a unique external IP address. This external IP address could be used, for example, to communicate to and from a fenced VM. Lab manager assigns a virtual router to route packets between the VMs inside the fenced environment. The virtual router is automatically configured when a fenced VM is deployed (launched) and deleted when the VM is un-deployed[5]. Another asset of the lab automation software is the fact that it is capable of running clustered type 1 hypervisor systems. According to IBM, a type 1 hypervisor provides its own operating system rather than being installed on an existing operating system. This arrangement enables type 1 hypervisors to achieve greater levels of efficiency, speed, security and reliability than type 2 hypervisors which rely on multi-purpose operating systems. Type 2 hypervisors are typically used in non-production environments where type 1 hypervisor's advantages are not as necessary[6]. Lab automation software products can control clusters of these more powerful and efficient hypervisors; as a result, the CPU and RAM load of the virtual machines can be spread across multiple servers with ease.

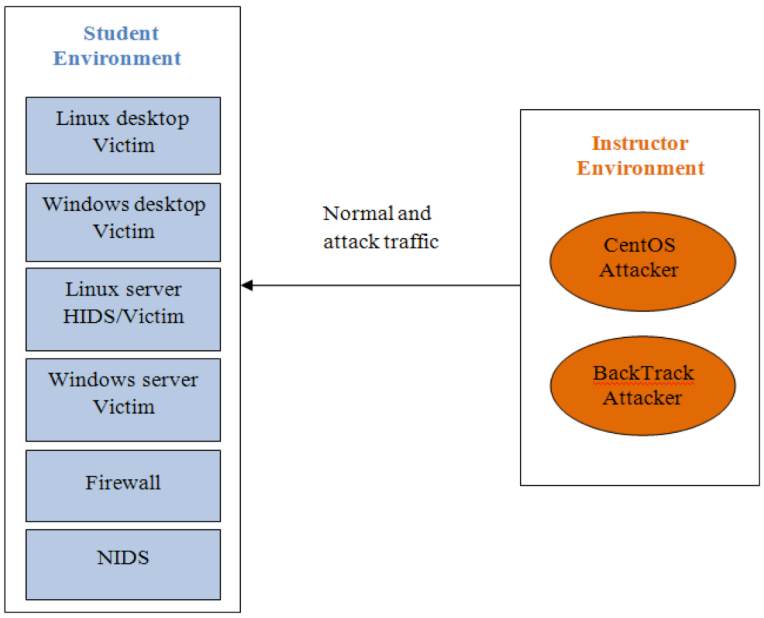

Figure 1. System Architecture

3. Experimental Environment

Security professionals never know when or how an attack is going to be carried out. In order to improve the learning experience, students should not know exactly when or how attacks will be carried out as well; this is why we believe the instructor should initiate the attacks rather than the student. The student may be required to take measures to prevent similar future attacks by possibly using a firewall or by incorporating an IDPS. The information provided by this research will afford students a more thorough understanding of IDPS and the environments they help protect; which in turn will provide the workforce with more capable security professionals.

3.1. Instructor Environment

Lab automation software allows for an immersive environment that can be configured to closely resemble a real world environment. In the process instructors could lauch four types of attacks.

- *Denial of Service* (*DoS*) attacks: Attackers disrupt a host or network service in order to make legitimate users not be able to have an access to a machine;
- *Probe* attacks: Attackers use programs to automatically scan networks for gathering information or finding known vulnerabilities;
- *User to Root* (*U2R*) attacks: Local users get access to root access of a system without authorization and then exploit the machine's vulnerabilities; and
- *Remote to Local* (*R2L*) attacks: Unauthorized attackers gain local access from a remote machine and then exploit the machine's vulnerabilities.

For example, Nmap[7] could be used to explore the services' statuses in the target machines of student environments. Guess password attack could be used to test the vulnerability of FTP server installed on the student environment. The instructors are able to:

- Launch attacks against student VMs from inside the environment
- Launch attacks against student VMs from outside the environment
- Launch attacks against student network and host IDS (HIDS/NIDS)
- Provide new VMs for attacks to be levied against during class
- Ability to view student's VM screen to help with assignments/issues

The instructor environment would be comprised of two Linux operating systems. One is the server-based CentOS Linux and the other is the appliance-based Backtrack distribution. These two operating systems will come loaded with penetration testing software. The instructor environment would also have a means of logging into one or more of the student operating systems in order to launch attacks from inside students' lab environments; this provides the means for an instructor to launch attacks against HIDS and NIDS based systems. The instructor would also have access to all of the student environments so that he or she could log in to any of the environments to help students or to launch attacks. Figure 2 shows the students' lab environments from the instructor's perspective. The instructor can launch attacks to all of the students' environments or to a specific student environment.

3.2. Student Environment

Utilizing virtual computing environments is not a new concept. It allows students to use their own personal computers to perform hands-on lab exercises. This flexibility allows students to work at their own pace and extends the lab environment to distance education students. In addition to the flexibility benefit of virtual computing environments, the use of lab automation software would add more value. For instance, it is possible for a student to control the environment with an internet connected hand-held mobile device. In fact, nearly any device capable of web browsing could control this environment as easily as a desktop computer.

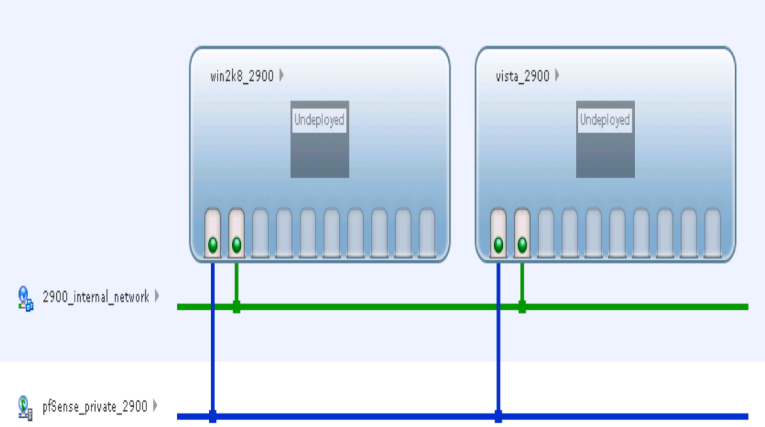

Figure 2. The Students'Lab Environment From Instructor's Perspective

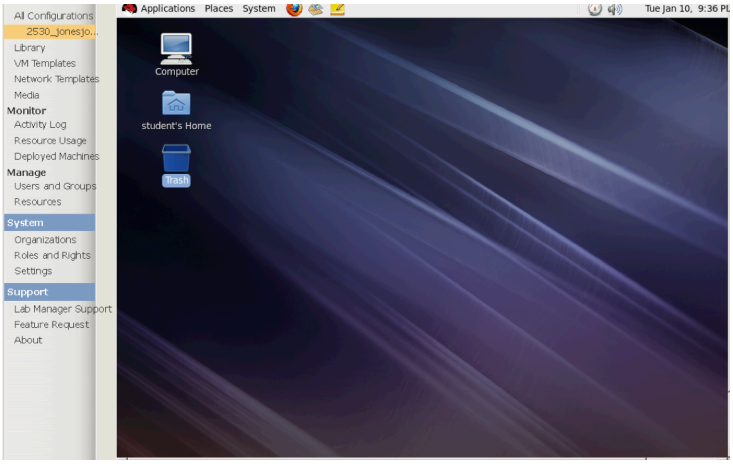

Figure 3. The Students'Lab Environment

A student would access the environment with his or her existing student network credentials, no need to remember an extra username and password. In addition, the environment would be exactly the same, no matter which computer students connected from; a student on vacation or out of town would not need to take a specific computer that contains the virtual environment because the environment exists in the cloud and can be accessed from anywhere. The environment would be completely set up in advance for the student on day one, there would be no need for the student to do any virtual machine or infrastructure configuration. If learning to configure the environment is academically important, students would still be able to configure whatever is necessary.

The proposed student environment would contain at least four multipurpose operating systems: two server based and two desktop based systems. One of the server based systems would host a syslog facility for centralized logging of the student's systems. One of the server based systems would contain a Host Based Intrusion Detection System (HIDS). A Network Based Intrusion Detection System (NIDS) would also be based in the student environment; this system may be

hosted on a multipurpose operating system or in a virtual appliance. Figure 3 shows the student's virtual machine. The student would be able to compare and contrast the attacks based on the feedback provided by the HIDS and NIDS based systems.

Students need to think from the defender side. Students need to recognize the latest of both attack and defense techniques. Then, students would understand how to apply a specific technique to prevent a certain type of attack. Also, students need to have the ability to write Snort[8] rules and analyze Snort log files and alerts. Therefore, once the attacks have been launched by the instructor, students can inspect Snort event output and therefore perform the correct corresponding contermeasure.

4. Conclusion

Lab management software has been in production use at East Carolina University for two semesters now, providing reliable service for a number of online courses. Though the current courses are not as complex as the proposed Intrusion Detection System course, research shows that the lab automation software would be an asset for such a use case. Our research indicates that a student could utilize the environment at the start of class without any setup effort required. The proposed setup would put four multipurpose operating systems and two virtual appliances in to the hands of IDS students; providing a far richer learning experience, an experience we expect would more closely resemble what would be found in a non-academic environment.

Bibliography

1. J. W. Ho, N. Mallesh, and M. Wright, "The Design and Lessons of the ASCENT Security Teaching Lab," Proceedings of the 13th Colloquium for Information Systems Security Education, pp.124-132, Seattle, WA, June, 2009.
2. W. Du, K. Jayaraman1, and N.B. Gaubatz, "Enhancing Security Education with Hands-On Laboratory Exercises," 5th Annual Symposium on Information Assurance (ASIA '10), pp.56-61, Albany, NY, June 2010.
3. L. Tao, L.C. Chen, and C. Lin, "Virtual Open-Source Labs for Web Security Education," Proceedings of the World Congress on Engineering and Computer Science (WCECS 2010), Vol. I, San Francisco, CA, October, 2010.
4. R. M. Cassado, *The Virtual Network System.* Special Interest Group on Computer Science Education, New York, 2005.
5. VMware, "How Does Fencing Work, VMware Lab Mananger," 2007. http://pubs.vmware.com/labmanager251/usergd/wwhelp/wwhimpl/common/html/wwhelp.htm?context=usergd&file=LM_Users_Guide_fencing.13.4.html.
6. IBM, "Virtual Systems Overview,"IBM Systems Software Information Center, http://publib.boulder.ibm.com/infocenter/eserver/v1r2/index.jsp?topic=%2Feicay%2Feicayvservers.htm.
7. Nmap. http://www.nmap.org (Last browsed in March 2012)
8. Snort. http://www.snort.org/ (Last browsed in March 2012)