2010

Xen Worlds: Creating a virtual laboratory environment for use in education

Benjamin Robert Anderson
Iowa State University

Xen Worlds: Creating a virtual laboratory environment for use in education

by

Benjamin Robert Anderson

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co-majors: Computer Engineering;

Information Assurance

Program of Study Committee:
Thomas E. Daniels, Major Professor
Doug Jacobson
Mani Mina

Iowa State University

Ames, Iowa

2010

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The Xen Worlds project uses the Xen hypervisor to create a virtual lab environment, providing students with personal networks of fully functional virtual machines (VMs) called a Xen World. The Xen Worlds environment can be provided using minimal hardware, and uses open source software, making it a low-cost option for education. The current hardware, consisting of five modest servers is capable of providing 470 VMs.

Since each Xen World can be isolated from each other, and from the Internet, students can be provided root access to their VMs without the security and privacy issues that would be present in a normal shared lab. In addition, to support off-campus students, Xen Worlds has several features that ensure the system is equally accessible and easy to use, even if the student has limited access to computing or network resources.

To rate the usability and effectiveness of the Xen Worlds environment, student feedback was collected through the use of surveys. The results indicate students feel the environment is an enjoyable and effective teaching method, with comments indicating a desire for a greater number of assignments to be provided.

CHAPTER 1. INTRODUCTION

Learning through practical experience has been an important concept for thousands of years, going back to ancient Greek philosophers.

"For the things we have to learn before we can do them, we learn by doing them."

– Aristotle

More recently, we see cognitive theories by Piaget and Vygotsky that link our interactions with the people and objects in our environment to our cognitive development and learning [Woo07].

Practical experience is recognized as an important part of providing students with the knowledge, skills and experience they need to be effective in the computer field. This can be seen in the integration of programming assignments and other lab work into many of the courses offered to students. Unfortunately, as students move into study areas such as operating systems, networks or computer security, they require full administrator access to the systems to access and modify the desired components of the system, raising many security and privacy issues [AD06].

In the past, there have been many efforts to create a safe, but fully functional, environment where students can experiment and learn without affecting other students in the class, or even the entire Internet, as happened with the Morris Worm [Mar90] . These efforts range from having computer labs dedicated to specific courses or purposes, such as the University of Calgary's virus and malware lab, to using stripped down, but functional, kernels such as Minix, or utilizing virtualization to isolate students within a virtual machine or network [AB04, AJD09].

The Xen Worlds project utilizes virtualization to create a safe, and functional lab environment, providing students with fully functional virtual machines (VMs) that use virtual network

bridges to configure them into arbitrary networks, (called a Xen World). The Xen Worlds are sandboxed from other, and from the Internet, providing a safe environment for student learning and experimentation.

Since Xen Worlds utilizes the Xen hypervisor as its virtualization layer, this approach does not require large amounts of computing resources. In fact, the Xen Worlds prototype was able to run 30 VMs on a single desktop computer [AD06]. In addition, Xen Worlds was designed to utilize a command line interface to the VMs, allowing students to remotely access their Xen World using SSH. This combination of remote access, and low bandwidth requirements, makes Xen Worlds suitable for classes that have on- and off-campus students.

There are many projects that utilize virtualization to provide a lab environment, and some have desirable features not provided by Xen Worlds, such as load balancing, student configurable networks and dynamically expandable hardware [KSR+05, AGR07b]. These extra features allow greater flexibility in hardware resources that can be utilized in building virtualization servers, and allow a network design component to be included with assignments.

However, these valuable features do so at the expense of ease ease-of-use for both instructors and students, requiring extra setup and installation effort. Using the additional features also adds complexity, and increases the time required for students to learn, and become comfortable with, the virtualized environment. Xen Worlds utilizes a simple, menu-driven interface for students, greatly reducing the time needed to learn the environment. This is especially important for off-campus students, as some will not have the option to visit the instructor or teaching assistant during their office hours to ask questions on any problems they are having. Also, these students may not have the option of discussing problems with other students in the class, except through Internet chat and forums, if those are provided for the course. In addition, Xen Worlds is designed to minimize the computational and network resources required to access the environment, allowing off-campus students without access to broadband connections, or hampered by high latency connections, to still utilize the environment.

Finally, Xen Worlds provides a collection of complete assignments - VM images, configuration files and assignment write-ups - that can be utilized by instructors adopting the Xen

Worlds environment, making it simple for instructors to include it in their courses.

In this thesis, we discuss underlying technologies and related work in Chapter 2, and then describe our implementation in Chapter 3. Chapter 4 describes the assignments created for our environment and how it maps to the IEEE-ACM Computing Curricula. Chapter 5 describes how we evaluated our environment, and presents the results. Chapter 6 presents our conclusions and describes future work.

CHAPTER 2. BACKGROUND AND RELATED WORK

In this chapter, we look at various types of virtualization technology, as the choice of virtualization, and Xen specifically, is an important aspect of the Xen Worlds project. In addition, we will discuss other projects that utilize virtualization for educational purposes, which will allow us to discuss the advantages, and drawbacks, of our selected approach.

2.1 Virtualization

Virtualization has been used in many different areas of computing to describe many similar concepts. In this thesis, we use the term virtualization as defined by Singh [Sin04]:

> "Virtualization is a framework or methodology of dividing the resources of a computer into multiple execution environments, by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service, and many others."

Therefore, we can use virtualization to divide the physical machine, called the *host*, into multiple environments, where each environment is used to execute a virtual machine (VM), or *guest*, running its own operating system. Thus virtualization allows for multiple VMs to execute on a single physical machine, but behave as if it were its own physical system.

While there are many approaches to providing this abstraction of the hardware, the general idea is that of the evil daemon proposed by Descartes [AJD09]. In Descartes' writings on philosophy, he describes a demon that is powerful and deceitful, and directs all their efforts to deceiving a person, creating an illusion of existence that the victim can not escape [Des88]. Virtualization behaves in a similar manner, as it utilizes an abstraction layer, which may be

integrated into the hardware itself, a hypervisor layer or even an application that "deceives" the operating system into believing it is running on native hardware.

In this section, we will examine the various technologies and software packages that provide virtualization, and discuss the advantages and disadvantages of their approach as they apply to an educational environment.

2.1.1 Virtualization Technology

The origins of virtualization can be found in the 1960s with IBM utilizing virtualization as a method of efficient time sharing for their mainframe computer systems [Ros04]. The initial approach taken by IBM is known as partial virtualization - where some, but not all, of the underlying hardware is emulated. However, later efforts did include full virtualization, where all aspects of the hardware were emulated, allowing for complete operating systems to run independently [Sin04]. However, with the decline of the mainframe due to the rise of the personal computer, virtualization was largely forgotten. However, due to the exponential increase in computing power, it has seen a resurgence, as multiple servers can now be consolidated onto a single physical host.

In addition to providing an abstraction for the underlying hardware, the virtualization layer must provide a variety of security functions. The two most important functions are to isolate the virtual machines from each other to protect the confidentiality and integrity of the instructions and data utilized by the VM, and to ensure the availability and fair use of the underlying hardware resources [Ros04].

The first function is particularly important in a colocation or cloud computing environment as different individuals and corporations, including direct competitors, may have virtualized systems running on the same host. Corporations using virtualization for server consolidation internally may require this separation for legal reasons such as Sarbanes-Oxley compliance or SEC rules. The second function prevents a software problem, (or malicious act), from claiming all of the resources on the host, starving other VMs on the same host.

There are a variety of general approaches to providing these virtualization services.

2.1.1.1 Full virtualization

Full virtualization is a complete emulation of the hardware through hardware, software or a combination of the two. Full virtualization has been utilized for many decades, and was implemented in the CP/CMS systems from IBM, (so named for the combination of Control Program and the Cambridge Monitor System)[Sin04]. However, it wasn't until 1998 that VMware managed to provide full virtualization for the x86 hardware [VMw07].

As a security feature, the x86 architecture provides 4 security rings, with the most privileged and trusted instructions executing in Ring 0, and the least privileged, (typically user programs), executing in Ring 3. Unfortunately, operating systems designed and implemented for the x86 architecture had the underlying assumption they were running directly on the hardware, and did not provide a method to trap privileged instructions executing in Ring 0. VMware worked around this problem by having the virtualization layer execute in Ring 0 and the operating system execute in Ring 1. This allowed the virtualization layer to perform binary translation on the privileged instructions, trapping them and converting them into safe instructions that can be executed [VMw07].

However, trapping and translating all instructions results in high levels of overhead and a more efficient method was needed. To solve this additional problem, VMware allowed unprivileged instructions from Ring 3 to be executed directly, eliminating the overhead for those instructions.[VMw07] For greater efficiency, VMware has moved to paravirtualization in its latest family of virtualization products, but QEMU is another product that provides full virtualization through the use of dynamic binary translation [Bel07].

2.1.1.2 Paravirtualization

Paravirtualization is implemented by defining a new hardware-software hybrid architecture, called a *hypervisor*, or *virtual machine monitor*, that is similar, but not identical to the underlying hardware. The main difference between the architectures is that the privileged instructions relating to CPU, memory and I/O are replaced with "hypercalls" to the hypervisor, while non-privileged instructions remain the same [VMw07, Nak07].

This approach greatly reduces the overhead inherent in dynamic translation, as the system calls no longer require the additional trap and translate process. Testing between native execution, hypervisor execution and other virtualization methods have shown the hypervisor approach is more efficient than other, software-based, virtualization techniques, in many cases being very close to native performance. A 2003 study performed six different benchmarks on various virtualization approaches and measured a worst-case slowdown of 8% for the hypervisor approach and a worst-case of 88% for other virtualization methods [BDF+03].

The major drawback of this approach is the operating system must be modified to run on the hypervisor architecture by replacing the privileged system calls with the appropriate hypercalls. In the Linux family of operating systems, this is done by patching the source code and recompiling the kernel. However, as this requires access to the OS source code, proprietary OSes, such as the Windows OS family, can not be ported to the hypervisor architecture. However, using device drivers ported to the hypervisor, provides some performance improvements over dynamic binary translation alone [VMw07].

The need to port the guest OS was eventually addressed for the x86 architecture when Intel and AMD added virtualization extensions to their hardware, allowing for hardware assisted virtualization.

2.1.1.3 Hardware assisted virtualization

While hardware support for virtualization was provided in many of IBM's mainframe systems, hardware support for x86 virtualization was not available until Intel and AMD included virtualization-specific extensions in their processors. When running with virtualization, privileged instructions are automatically trapped in the hardware and redirected to the hypervisor [VMw07]. Since the hardware itself traps the instructions, no modifications to the guest OS is required.

The VT-x and AMD-V extensions also provide new virtualization and security functions that can be utilized by the hypervisor to efficiently protect and manage the guest systems [Dev07, Cor08]. This allows the hypervisor to efficiently handle and translate instructions

from the VMs without the high overhead of software-based binary translation, and without the need to port the guest OS to a virtualization architecture. This allows closed-source operating systems to be efficiently run in a virtualized environment.

While hardware assisted virtualization requires hardware containing the necessary extensions, these extensions are provided with most processors available from Intel and AMD.

2.1.1.4 Operating system-level virtualization

Unlike the previous methods discussed, this approach to virtualization moves the virtualization functions into user-space. By creating isolated containers, (called *jails* in FreeBSD), untrusted users and applications, including an OS, can operate in separated, secure environments without creating a risk to each other or the underlying system. In User-Mode Linux (UML), the ability to run virtual machines was created by altering the kernel to run on itself utilizing only user-space processes; i.e. - UML is the guest, and the host OS [Dik00].

FreeBSD jails provide partial virtualization at the operating-system level by providing each jail with its own file system, processes and IP address - and providing a root user. However, the kernel is modified to prevent any actions by the root user that would allow it to break out of its jail system [Pro09]. Examples of these prohibited actions are: loading or modifying kernel modules, adding devices and altering network adaptors, as these require modifications to the underlying system, breaking the paradigm. However, regular users of the FreeBSD jails will not be able to tell they are running in a jail environment [hKW00].

This approach eliminates the overhead of binary translation described above, but it does have greater overhead than other approaches as privileged instructions must be translated to less efficient user-space calls.

2.1.2 Advantages and Disadvantages of Virtualization

The different approaches to virtualization have their own advantages and disadvantages, but we must also consider the advantages and disadvantages to the overall concept of virtualization. In this section, we will examine the main advantages and disadvantages of the virtualization

approach including their impact an educational environment.

2.1.2.1 Advantages

The first advantage of virtualization is it allows for efficient use of hardware through the consolidation of various systems onto a single physical host. This allows for lightly used servers to be migrated onto a single physical host, reducing hardware costs, administration time and operational costs such as floorspace, power and cooling. The cost saving can be non-trivial, with VMware claiming virtualization can:

> Reduce hardware and operating costs by as much as 50% and energy costs by 80%, saving more than $3,000 per year for every server workload virtualized [VMw09].

Consolidation is a very important feature for educational institutions, particularly smaller schools, that may not have the resources for extensive labs. In addition, educational institutions may want to provide many different environments to give students a wider variety of experiences. Virtualization allows a single host to run different OSes, such as Linux and Windows, and have hosts running different applications such as providing both Apache or Internet Information Services web servers [AJD09]. Consolidation also reduces the administrative load on computer support staff.

A second advantage of virtualization is it provides for sandboxing of the VMs. One way to benefit from this feature is to create a new VM when executing any untrusted application, as any damage would be limited to the single VM, and not the underlying host [Sin04]. It is also important in hosting facilities, such as Amazon's E2 cloud services, since there is no guarantee systems from competitors are not being hosted on the same hardware. Since small companies with little or no internal IT staff are particularly able to benefit from utilizing cloud services, it would be impossible for a server provider, such as Amazon, to be aware of their local competitors. Even internally, a company can benefit from this isolation as Section 404 of the Sarbanes-Oxley act requires "internal control over financial reporting", and must assure the integrity of any servers related to company finances or reporting [SC07].

Sandboxing is particularly important in an academic environment as students are learning to use and modify functionality they may have little or no experience with, that may result in critical failure of the system if errors are made. However, it is much easier and faster to restore a VM than it is a physical machine [AJD09]. In fact, many virtualization applications allow checkpointing of the VM images, allowing students to quickly roll-back to a previous snapshot and continue their work.

Sanboxing is also advantageous when compared to the shared lab situation common for most courses. In a corporate environment, access to servers is generally restricted but, in an educational environment even labs restricted for use in certain courses are accessible to all enrolled students. With some labs requiring administrator permissions to perform assigned tasks, it is possible for malicious students to install a rootkit or key logger to steal another student's work, or steal passwords [AJD09].

There are also some disadvantages to utilizing virtualization instead of independent physical systems.

2.1.2.2 Disadvantages

The most obvious disadvantage is creating a single point of failure, (the host system), for all of the guest systems. While this can be mitigated by using some of the hardware cost savings to purchase enterprise grade hardware with reliable and redundant components, any failure in the host OS, networking or hardware can impact dozens of systems.

Another problem with virtualization is a reduction in ability to handle spikes in CPU or network usage. If the number of VMs running on the host is determined by the average expected CPU load, spikes in usage may require more resources than available, resulting in guest systems having degraded service. This has been seen in a break-in lab assignment, where numerous students ran a password cracker for extended periods of time, overloading the CPUs, and creating sluggish behavior all of the VMs on the host.

However, when compared with the benefits of virtualization, there are many business and educational situations where there is a clear case for utilizing virtualization.

2.2 Cloud Computing

Cloud computing is an important paradigm shift occurring in computing where hardware and software services are offered as a utility from large datacenter providers. A Berkeley technical report offers the following definition of cloud computing [AFG+09]:

> Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. The services themselves have long been referred to as Software as a Service (SaaS), so we use that term. The datacenter hardware and software is what we will call a Cloud.

By definition, these services are billed on a pay-as-you-go model, as rental payment models would be considered belonging to a colocation model of computing. However, there are generally pricing advantages if customers are willing to guarantee certain running durations and levels of service for their VMs. In Amazon's Elastic Computer Cloud (EC2), customers willing to guarantee a one-year term with a default instance can, with a one-time fee of $227.50, reduce their usage fee for Linux/UNIX systems from $0.085/hour to $0.03/hour [Ser09].

In order to be economically viable, providers must ensure efficient use of their hardware resources which is accomplished through the use of virtualization. However, this can lead to a trade-off between efficiency, which benefits the provider, and flexibility, which benefits the customer. In an attempt to provide both, EC2 utilizes the Xen hypervisor, which provides efficiency for systems ported to the Xen architecture, but still allowing for other operating systems to run using hardware-assisted virtualization. However, to offset the inefficiency in using non-ported operating systems, (such as Windows), EC2 charges $0.085/hour for a default instance of a Linux/UNIX OS, but $0.12 for a Windows-based OS [Ser09].

There are many advantages to cloud computing to educational institutions seeking to implement a virtual lab environment. First, cloud computing eliminates the capital expenditures required for the lab environment as all hardware is hosted by the service provider. Even though virtualization provides a greater ability to efficiently use hardware, being able to eliminate all

hardware expenditures is an important consideration. In addition, using the cloud eliminates the overhead of installing and maintaining the host systems.

Another important consideration in utilizing the cloud is the elimination of a single point-of-failure, and the inability to handle spikes in CPU or network usage. Economies of scale dictate the cloud be provided from very-large-scale datacenters, meaning redundant and backup systems that can ensure high-availability. In addition, one of the primary purposes of the cloud is to provide elastic computing, where additional resources can be brought in on an as-needed basis. This allows the graceful handling of any CPU or network usage spikes, although it may result in higher operation costs.

Finally, another important advantage of utilizing the cloud in an academic environment is the ability to pay for the cloud usage through student lab fees. While it may be difficult to determine appropriate lab fees for equipment that will be used and maintained over several semesters, and the difficultly of budgeting for unknown administrative overhead and maintenance, it is fairly simple to determine the cost for cloud services for a given semester. Given the rates for the EC2 service mentioned above, the cost for a semester of cloud utilization could be less than $100, making it less expensive than many textbooks.

2.3 Related Projects

Given the importance of integrating lab assignments into computing classes, and the benefits of virtualization, it isn't surprising that there are many projects dedicated to creating virtual laboratories for educational use. In addition, from a pedagogical view, it has been shown that utilizing a virtual lab environment results in the same student performance as utilizing a physical lab and that students enjoy using a virtual environment as it allows them to work on their projects from home [LS06, DTW07].

In this section, we examine four projects that utilize virtualization to provide their lab environments, selected for their focus on simply providing functioning systems and networking to students. While there are many other projects such as the Virtual Distributed Ethernet project, the Open Network Laboratory project and PlanetLab utilize virtualization, but are

Feature	Xen Worlds	SEED	SOFTICE	V-NetLab	Capstone
Virtualization Technology	Xen	VMware	User Mode Linux	User Mode Linux/Xen	VMware
Student Defined Networks	No	Yes	Yes	Yes	Yes
Dynamically Scalable Hardware	No	No	Yes	No	No
Student Interface	Menu-driven	VMware Client	SSH/X-Windows	SSH	VMware Client
Assignment Library	Yes	Yes	No	No	No
Open Source	Yes	No	Yes	No	No
Need To Learn Environment	No	Yes	Yes	Yes	Yes

Table 2.1 Virtual Lab Environment Features

primarily focused on the network or low-level behavior, so are not reviewed [AJD09]. It is also important to note that there are no known large-scale projects that utilize the cloud to provide a virtual lab environment. While there are some smaller projects that have utilized a "local cloud" paradigm, there has not been a large move to a large-scale cloud service such as Amazon's EC2 [MM09].

For each reviewed project, we will describe the methods used to host VMs and to create virtual networks, hardware and software utilized to create, manage and run the hosts and guest systems, additional features, and how students access and interact with the environment. A matrix comparing some of the features of the different approaches can be seen in Table 2.1.

2.3.1 SEED: SEcurity EDucation

The SEED project, (SEED from SEcurity EDucation), is a virtual lab environment, and group of assignments for computer security education [DTW07]. One of the primary motivations for SEED was to provide a common environment for lab exercises in multiple areas of computer security [DW08]. To utilize this environment, students use VMware, (or Virtual PC), to host VMs running the Linux or Minix operating systems. This provides flexibility as labs requiring in-depth examination of kernel modules are simpler in Minix, while labs requiring a full-featured OS can use Linux. An important aspect of this software is that Linux and Minix are both open-source software solutions, and VMware is available for free for educational use. However, since VMware supports a variety of other OSes, students have been able to complete labs using non-standard, (for the SEED environment), operating systems such as FreeBSD [DW08].

Since all of the software is free, students can download and install the required files and

software on their own machine, eliminating the need for a public lab. However, the software is also provided in public computer labs, so students have the choice of where to work. Unfortunately, due to the large storage requirements for VMs, which can be 1 GB or more in size, VM images cannot be stored on lab systems [DTW07]. However, given the low cost of USB flash drives, students can easily store their own VMs, and transport them wherever they wish to work on their assignments.

Virtual networks can also be constructed as VMware supports the creation of various LAN environments. These networks can support heterogeneous operating systems in arbitrary network topologies, and can be routed through the Internet if desired [DW08].

In addition to providing a common environment for lab exercises, the SEED project also aims to provide a library of assignments, covering a variety of areas, that can be utilized by other institutions. Currently, SEED has developed 12 assignments that can be grouped into the categories [DW08]:

- Design and implementation: Where students design and implement security modules.
- Exploration: Provides the students with the opportunity to analyze systems.
- Vulnerability: Where students exploit vulnerabilities or create countermeasures.

Feedback through student surveys has shown that students enjoy the SEED labs, and the environment, illustrating the value of this approach [DTW07].

2.3.2 SOFTICE: Scalable, Open source, Fully Transparent and Inexpensive Clustering for Education

The SOFTICE project was created to provide an environment where student had access to their own systems and could create networks of any size, in any topology [AGR07b]. To create this environment, SOFTICE addressed the problem in three areas [AGR07a]:

- Virtualization
- Remote Access

- Scalability

For their virtualization needs, SOFTICE utilizes User Mode Linux to provide a large number of VMs to students, and sandbox the student machines. For access, students use SSH or X-Windows to connect to a "master node", that acts as a gateway and stepping stone into the environment. Finally, scalability is addressed by utilizing a cluster of compute nodes created using older PCs and the Warewulf clustering toolkit [AGR07a]. One of the benefits of using a Warewulf cluster is it supports the dynamic addition of hardware, and automatic load-balancing, so the cluster can be expanded as needed. As all administration of the compute nodes is handled by the Warewulf toolkit, no additional administration of the additional machines is required.

A pedagogical goal of the SOFTICE environment is to minimize the need for students to "learn the laboratory platform", so they can maximize their time with the environment [AGR07a]. To reduce the time needed to learn the environment, students use the Manage Large Networks (MLN) software to create their VMs and networks. Once students connect to the master node and upload their MLN configuration, the VM images are pulled from the SOFTICE file server, and the VMs and networks are created. Networking is achieved through the use of UML switches, which can also be set to act as a network hub [AGR07b].

To increase efficiency, MLN also provides support for Copy on Write (CoW) filesystems. VM images can be 1GB or more in size, however, different VMs may only differ in a few key files. CoW allows the system to store a single read-only image to serve as the base image for a VM, with a smaller read-write image that contains only the modified files. This can greatly reduce the storage space required by the system.

2.3.3 V-NetLab: Virtual Network Laboratory

V-NetLab was developed to provide a safe and isolated educational environment for student in computer and network security courses [KSR+05]. In addition, it was designed to address the isolation problems with conducting research on self-propogating malicious code and other, potentially harmful, applications [SKKS08].

For educational use, three challenges were identified [KSR+05]:

- Configuration changes may damage the entire OS.
- Students may misuse administrator or root permissions.
- Costs associated with providing a physical laboratory.

To address these issues, V-NetLab uses VMware to provide VMs running Linux or Windows OSes. These VMs are load-balanced on a cluster of workstations, with a single NFS server providing the storage of the VM images [SKKS08].

V-NetLab supports the creation of virtual networks through their novel datalink layer traffic translation. All network traffic is encapsulated by their protocol, ensuring undesirable traffic, such as that generated by malicious code, cannot escape the environment [SKKS08]. In addition, this translation can allow all higher layer protocols, such as TCP/IP to be visible to the VMs. (Or, if desired, hidden.)

To access the V-NetLab environment, and their virtual networks, students use SSH to access a gateway system, and using it as a stepping stone to access their VMs and virtual networks. The gateway machine enforces security as students are only able to connect to their own VMs, maintaining the isolation provided by VMware. The gateway also allows students to transfer data in and out of the environment using Simple File Transfer Protocol. However, without the explicit construction of a connection, the gateway blocks all communication between the V-NetLab environment and outside networks [KSR+05, SKKS08].

Students are able to define their own networks through configuration files that define the VMs, IP addresses, network devices and any startup scripts to be executed on the machines. The system then performs checks to ensure a valid configuration and registers the network. Once registered, the student can run and access their VMs and virtual networks. Another alternative is for the instructor to create and pre-register a network for all students to utilize, although each student will have their own individual instance of the network [KSR+05].

2.3.4 Iowa State University: Information Assurance Capstone Course

The virtualized environment utilized for the ISU information assurance capstone course was created for 2 purposes. The first was to leverage experiences with cyber defense competitions to create a graduate course based around the same concepts and, second, to provide off-campus students a way of completing the creative component of their Masters degree without requiring them to travel to the ISU campus [EBJ09]. In essence, the class is an extended cyber defense competition that takes place on virtualized systems and networks.

The capstone environment utilizes VMware ESXi as the hypervisor and utilize the VMware networking tools to network the VMs. As this is a capstone course for a graduate program, students are required to design and implement all aspects of their own networks including OSes, network topology and the location of any security systems, such as firewalls [EBJ09]. The actual environment is isolated from the Internet by a host machine that acts as a firewall and stepping stone between the Internet and the capstone networks.

To create their VMs, networks, and manage them, students utilize the VMware Infrastructure Client, connect to the host machine, and then use the admin functions of the Infrastructure Client to manage their networks. During the class, students will then operate from these VMs and networks to defend their systems, and conduct attacks against other students in a "Capture the Flag" competition that lasts for several weeks [EBJ09].

CHAPTER 3. IMPLEMENTATION

In this chapter, we examine our implementation of the Xen Worlds environment, beginning with the goals and requirements that guided the creation and evolution of the Xen Worlds project. In addition, we will examine the hardware and software utilized in the environment, how it evolved over time, the motivations behind these changes, and any lessons learned from our experiences.

3.1 Goals and Requirements

The Xen Worlds environment was created to provide a safe, lab environment for use in the Information Assurance (IA) graduate program at Iowa State University. The IA program covers a variety of topics, such as hacking techniques and advanced network concepts, that would require root access to the provided systems, raising a large number of issues regarding the security of the systems themselves, and the privacy of the students using them.

Prior to the Xen Worlds project, the only option available for laboratory assignments was a physical lab, dedicated to specific courses, with all of the resources shared by the students. However, this meant any crashes or system problems that was the result of one students work, could prevent the entire class from accessing the lab systems until the systems could be rebooted or restored. In addition, students were not given root privilege to these machines, and were limited in the type of activities they could perform.

The goal of the Xen Worlds environment is to provide a safe environment where students can configure systems, explore networks, and conduct attacks without any risk to other student systems, or the campus network. In addition, because many vulnerabilities are system or service specific, the environment had to accurately reflect real-world systems. However,

achieving this in a physical lab would be cost prohibitive, so it was decided the environment would use virtualization to eliminate much of the overhead.

In addition to traditional, on-campus students, the IA program offers an online option for off-campus students and, in several classes, they outnumber the on-campus students. Off-campus students may have to operate under many constraints that on-campus students do not. For example, all of the following issues were encountered:

- Students did not have access to broadband connections, or experienced high latency.
- Utilized corporate computing resources, and did not have the ability to change operating systems, install new programs, or send certain types of network traffic (i.e. - encrypted).
- Resided in a different time zone, (such as Iraq), introducing hours of delay for answers to any questions, greatly slowing progress.
- Worked at irregular hours due to full-time employment and other professional commitments.

Finally, since there was no immediate funding for the project, a low-cost approach to hardware, software and administration was vital.

Given the above, the following requirements was established for the Xen Worlds environment[AJD09]:

Technical Requirements

- Utilize free or open source software and operate on low-end hardware.
- Minimize administrative overhead.
- Support a variety of complete operating systems.
- Support arbitrary sizes of networks and topologies
- Sandbox the VMs from the Internet and other students.
- Scale to support a large number of students and VMs.
- Support 24/7 access to the environment.

Pedagogical Requirements

- Provide similar interaction experience for on- and off-campus students.
- Ensure ease of use for students, instructors and administrators.

3.2 Hardware

One of the important constraints on the environment, was the assumption off-campus students may not be able to install software on their personal systems. This assumption must Xen Worlds store all of the VM images locally, and provide the computing power to run all of the VMs concurrently. (It was assumed the system would be highly utilized as assignment deadlines approached.) There have been several different approaches utilized in the underlying hardware for the Xen Worlds project, with changes being made based on available funding and scalability issues that were encountered. These resulted in 3 distinct phases of underlying hardware, ignoring minor upgrades to memory or storage upgrades during a phase. These phases are:

- Desktop prototype
- Diskless cluster
- Dedicated enterprise-grade servers

In this section, we will describe each phase of hardware development, the number of VMs supported and any encountered issues that are specific to that approach.

3.2.1 Desktop prototype

As mentioned previously, the Xen Worlds project was started without funding, requiring the prototype to be created using a desktop system already present in the graduate office. The desktop system had a Pentium 4 processor and 2 GB of RAM with 100 GB of hard drive space running the Red Hat Enterprise Linux Workstation v.4 operating system with Xen 2.0 installed [AD06].

This system was capable of supporting up to 30 concurrent VMs, with the memory requirements for each VM being the limiting factor. This hardware approach was very effective, but could not scale to provide the number of desired VMs to each student in a class.

3.2.2 Diskless cluster

The second phase of hardware development was designed to address the scalability issues encountered with the prototype, while still keeping costs to a minimum. In this phase, a cluster of 8 diskless computers were utilized to provide the computing and memory required to run large numbers of VMs. These systems, (called boards since they did not have a case, and the motherboards were directly mounted to a small metal rack), had a Celeron 2.0GHz processor and 2 to 4 GB of RAM. The metal rack also contained a single 1U system, which allowed the boards to PXE boot, provided the storage for the VM images and acted as the gateway between the cluster and the campus network. Communication between the 1U and boards utilized Gigabit Ethernet, with a 100 Mbit Ethernet connection to the campus network. This hardware configuration supported 300 concurrent VMs [AD06].

The first problem encountered with this solution was the difficulty in performing a PXE boot with the Xen architecture. Since the kernels being loaded were ported to the Xen architecture, the sanity checks within PXE caused a kernel panic due to an unknown architecture. This problem was solved by using a boot loader that supported the Xen architecture, but took several weeks of investigation to identify a solution.

The second issue was the reliance on a single storage location for the VM images. During a mass startup of the VMs, the large number of files being accessed, by a large number of VMs, resulted in disk thrashing, greatly reducing performance. However, while this caused large delays during startup, once the VMs had finished booting, the system entered a well-performing steady state.

The final issue with this hardware approach was the large amount of storage required to store several hundred VM images, and the bandwidth required transfer those images over the network. While a hard drive was added to one of the boards to offload the VM storage from

the 1U, it was found the cluster network was heavily loaded, and could easily become saturated if usage expanded. Fortunately, funding for enterprise-grade hardware became available, and allowed the move to the current hardware approach.

3.2.3 Dedicated enterprise servers

The current phase of hardware utilizes enterprise-grade servers to host, store and run student VMs. The current Xen Worlds "server farm" consists of 5 Dell PowerEdge servers with dual-processor, dual-core Xeon 1.60GHz processors, 16 or 32GB of RAM, and 2TB of hard drive space with integrated RAID controller in each server [AJD09]. In addition, the processors have the Intel VT-x extensions, allowing full virtualization if desired. Using just three of these systems, Xen Worlds has been able to run over 200 VMs, with a current maximum capacity of 470 VMs if all 5 servers are utilized. This VM limit results from Xen's naming scheme for block devices, and not the hardware so the maximum possible number of VMs that can be concurrently run is still unknown.

Currently, there are no issues with the current hardware approach but integrating the servers into a single logical cluster, similar to a cloud, is under discussion. However, with the current system meeting the level of demand and service desired, there are no plans to implement any changes.

3.3 Software

While there were many virtualization approaches that would have met many of the requirements listed above, the need to minimize costs and scale well on low-end hardware led to the adoption of the Xen virtual machine monitor as the virtualization provider. Xen is open source software, so it was free to use, and also enjoyed excellent performance compared to other existing virtualization solutions [BDF+03].

3.3.1 Operating Systems

Xen Worlds utilizes two different operating systems for the host machines - Red Hat Enterprise Linux (RHEL) and Fedora 8. The Fedora Linux distribution is Red Hat's "bleeding edge" development distribution, and is their test and development environment for the RHEL distribution. For this reason, Fedora does not always have the features or support expected in a commercial operating system. For example, Fedora 8 is utilized since follow-on releases of Fedora dropped native support for running as a host. While it is still possible to configure as a host system, (by patching and compiling from the Xen source code), the need for overall ease-of-use during installation keeps Fedora 8 as the open OS of choice.

RHEL is the commercial operating system provided by Red Hat. While it is more robust and better supported than Fedora 8, customers must purchase a support plan to obtain support and updates. Unfortunately, adding the Virtualization "channel" needs to be done as a system update, so can not be used as a completely free solution. Fortunately, Iowa State has purchased a site license for RHEL, and systems registered through the campus are able to apply the updates. While RHEL is more stable and better supported than Fedora 8, and makes for a better OS choice when possible, to ensure Xen Worlds can be a low-cost approach, both operating systems are used and supported.

3.3.2 Xen Worlds Middleware

To simplify the creation, administration and access to the Xen Worlds, custom middleware was written to simplify these tasks. The middleware is written in Python, and has been released under the GNU GPL. The middleware, and an installer, are available from the Xen Worlds project website [And09]. If required, the installer will install the required Expect and Pdmenu packages prior to installing the Xen Worlds middleware.

The middleware utilizes two configuration files, a world configuration file and a VM-specific configuration file, which are described in more detail in the Instructor Interface section. The middleware utilizes the world configuration file to create the required VM images, Xen configuration file and network interfaces, and utilizes a naming convention to avoid file or virtual

network adaptor name collisions between students.

Configuration of the VM is accomplished through the use of an Expect script, generated from the VM configuration file. Originally, the VM was configured by mounting the image on the host system, and configured the system by writing to the configuration files. However, as Xen matured, it moved from a flat file for the VM image, and used a drive image file with multiple partitions. While the appropriate partition could be mounted and configured, errors unmounting the filesystem were common, and could result in filesystem errors. To handle this issue, and to prevent problems in the future, the configuration operations were modified to utilize Expect scripts.

The middleware parses the VM configuration file, performs replacement operations on any metatags within the file to generate the Expect script. The VM is started with the default configuration and, once booted, Expect is executed to log in, perform operations and restart the VM with the new configuration. In addition to eliminating the difficulties with mounting the filesystem, utilizing Expect allows for operations that don't utilize plain text configuration files, or commands that perform multiple tasks may not be easily accomplished with direct manipulation. While startup scripts could be generated, and placed on the VMs to execute, this is simply moving the method in which the script is executed.

3.3.2.1 Instructor interface

To define a Xen World, the instructor creates a world configuration file which, consisting of three sections:

- World Data - Provides the number of nodes, number of world-specific networks and any additional network interfaces that will be common among all Xen Worlds.
- Node data - Indicates the VM image that will be the base image for the VM, and VM specific attributes such as memory, subnets and IP addresses.
- Network Data - Lists the nodes that are going to be connected to a specific network.

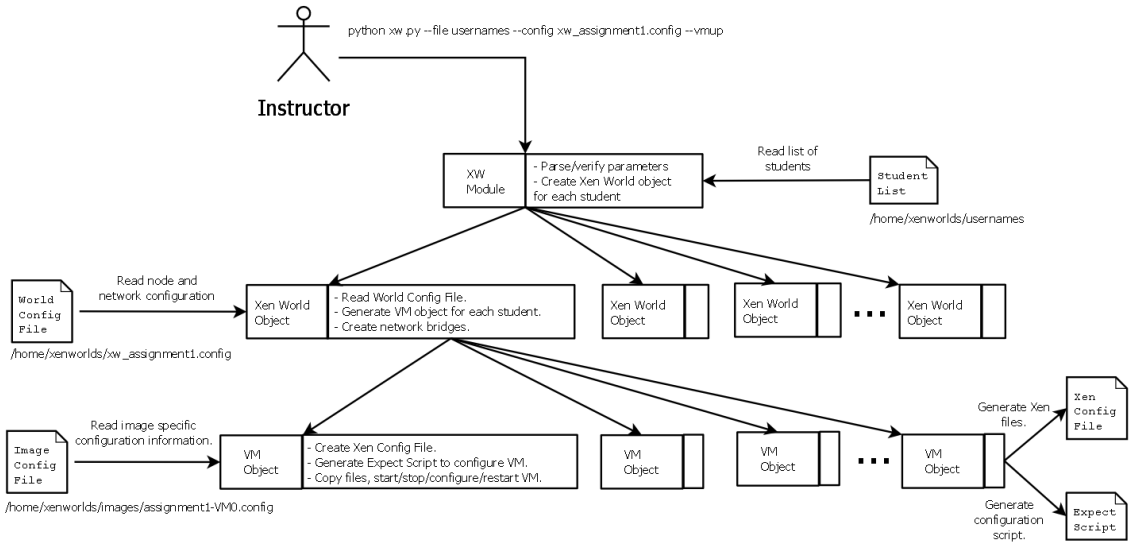

Figure 3.1 Instructor interaction with the Xen Worlds middleware

A second configuration file, specific to each VM image, is required to define the required configuration commands and configuration file locations. This eliminates the need for Xen Worlds to be aware of OS specific commands and configuration methods, simplifying support for multiple OSes. For example, the hostname for Linux is defined in the /etc/sysconfig/network file, while FreeBSD defines the hostname in the /etc/rc.conf file [Pro09]. This file can contain metatags for items such as IP address, or network adaptors so they can be replaced with the appropriate values given in the Xen World configuration file.

Once defined, the instructor can create and configure the Xen Worlds for a single user, or a list of users, by executing the middleware with the appropriate flags. The command to create and configure a Xen World for every user listed in the file usernames would be:

```
python xw.py --file usernames --config xw_assignment1.config --vmup
```

How that instruction would interact with the Xen Worlds middleware can be seen in Figure 3.1.

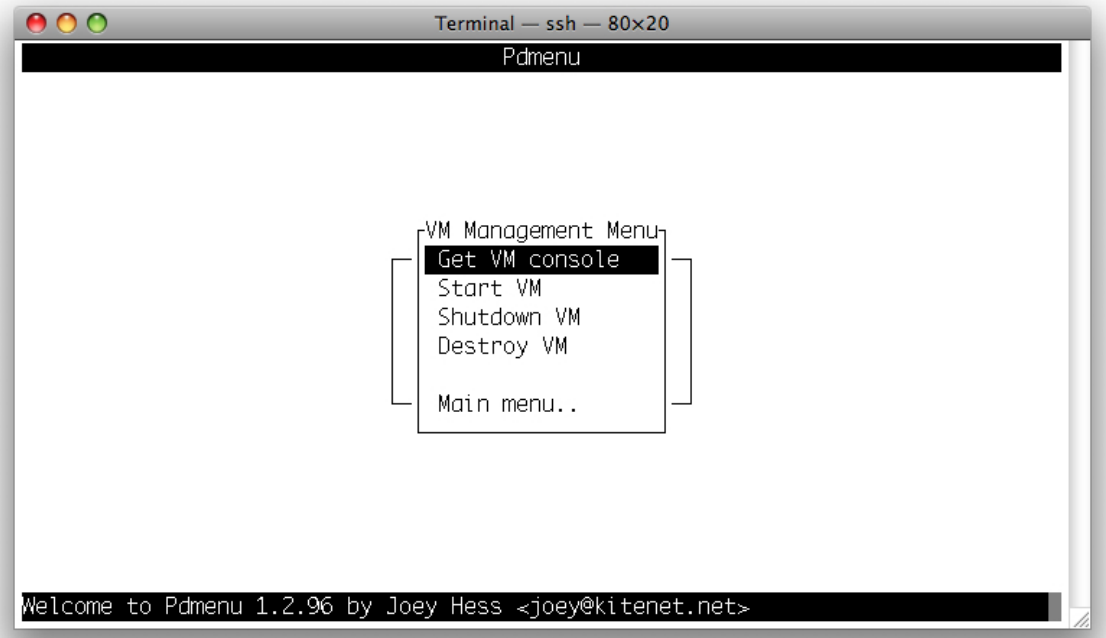

Figure 3.2 Student interface to the Xen Worlds environment

3.3.2.2 Student interface

The student interface to the Xen Worlds environment is through Pdmenu, (a menu-driven shell), which is accessed through an SSH client [Hes08]. A menu-driven interface was selected as being being the most user-friendly as a menu interface can reduce the users memory load, promote exploration and avoid certain types of syntax errors [LR93]. In addition, a menu-driven interface eliminates the need to learn the underlying administrative functions and syntax and increases the security of the host system as it only allows students to execute pre-defined commands. In Xen Worlds, students are restricted to VM management commands, and the passwd command in order to change their own password. A screenshot of the interface can be seen in Figure 3.2.

The use of SSH, and utilizing a command-line on the VM requires minimal bandwidth, and is easier to use with severe lag, allowing a similar experience for on- and off-campus students. If required, the menu can also be accessed via Telnet so individuals working from a business

that does not allow encryption on outbound connections can still access the environment.

CHAPTER 4. ASSIGNMENTS

In this chapter we present the lab assignments that have been developed for the Xen Worlds environment, and are currently part of the assignment library. For each assignment, we will provide a brief description of required tasks, the educational goals of the assignment, our experiences with these assignments and the areas of the IEEE-CS and ACM Computing Curricula 2001 Computer Science (CC2001) addressed by the assignment

4.1 Authentication

In this lab students are provided with a Xen World consisting of 2 networked VMs, and the teaching assistants (TA) public key. Students are required to configure VM0 to accept root logins based on the TA's key, generate their own public/private key-pair, and configure VM1 to accept root logins from VM0 using the generated key-pair.

While a relatively small lab exercise, this assignment is generally used to familiarize the students with the Xen Worlds environment. In addition, it demonstrates an authentication method other than the standard username/password combination, reinforcing how public and private keys can be utilized.

In our experience with this lab, the largest issue was the transfer of the TA's public key to the VM. Since file transfers are not allowed into the environment, students have to copy-and-paste the key into their SSH session. However, many students also used an intermediary application between the course website and the SSH session, (such as Microsoft Word), which could introduce invisible character such as page breaks, or other formatting characters. This would then corrupt the key, making logins with the TA's private key impossible. In this situation, allowances were made since there was no way for students to test their solution with

the TA's private key.

CC 2001: AL9 - Cryptographic algorithms, OS7 - Security and protection, NC3 - Network security.

4.2 SSH Address Harvesting

For this lab, students are provided the root password for a VM operating as a gateway router for a fictional company named PirateSoft. The PirateSoft network consists of 17 VMs, including the gateway, and are divided into 6 subnets. Using SSH and utilizing the IP addresses stored in the known_hosts files, and keys stored in the authorized_keys files students travel through the network searching for certain pieces of information regarding a secret company project with the codename: Black Pearl. In addition, since the focus is on traversing the network, and not searching for files on compromised machines, an email "clue" file is left on one of the machines, detailing the hostnames and directories where the project information is stored.

The purpose of this lab is to reinforce the ideas found in the paper *Innoculating SSH Against Address Harvesting* that describes utilizing information cached by SSH, specifically the known_hosts and authorized_keys files, to compromise an entire network [SJSM06].

Experience with this lab has been very favorable, as students seem to enjoy taking on the role of attacker. The largest issue is students becoming "lost" in the PirateSoft network as there are cycles within the network that can cause students to lose track of what VM they are currently operating on. However, after becoming lost, students generally turn to the creation of a network map to understand where they are within the network.

CC 2001: NC3 - Network security, SP8 - Computer crime.

4.3 Software Security

Students are given 3 VMs, networked in series, and are given the root password to the middle VM, which is connected to both LANs. The other VMs have unknown root passwords, but are running "remote access servers" that contain numerous, exploitable bugs. For one

of the servers, students are given the source code to a client that can access the server, and the server code itself. For the other server, which uses a pseudo-random port, username and password - and additional security checks not detailed in the source code from the first server, students are only given an executable client. Students are also provided with John the Ripper, a password cracking application. The goal is for students to gain root access to the 2 VMs by any means possible.

The purpose of this lab is to reinforce topics in software security by having students review source code, locate vulnerabilities and exploit them utilizing the client application. The hardened server also requires students to perform black box analysis of the code to determine the port, username and password by monitoring the network traffic generated, and any error messages output by the hardened server. This provides students with a more realistic penetration testing experience.

The biggest problem with running this lab is the potential for students to corrupt key system files in their efforts to gain access, that are not corrected through a simple reboot. In this situation, depending on system damage, the TA may repair or replace the VM. However, since students operate on their own VMs, this kind of outage does not affect other students.

CC 2001: OS7 - Security and protection, SE1 - Software design.

4.4 Firewall

As with the previous lab, students are provided with 3VMs, networked in series. However, students are given the root password to all of the VMs. The goal of the lab is to use iptables to turn the central VM into a firewall that implements a given security policy.

The goal of this assignment is to have students understand the different ways network traffic can be filtered, and to become familiar with rule chains, where a variety of rules are applied in a specific order. Our experience with this lab is students are able to perform the operations with minimum assistance, probably due to a large body of tutorials that are available on the Web.

CC 2001: NC6 - Network management.

4.5 Access Control

This lab requires only a single VM. Students are provided with the details for a fictional company that includes:

- Employee lists
- Corporate departments and hierarchy
- Desired directory structure
- Security policy

Using this information, students are required to configure the system to include accounts for all users, generate all necessary groups and any other actions required to enforce the provided security policy. In addition, the security policy has several entries that conflict with each other, and some that cannot be implemented using standard Unix permissions.

There are three educational goals for this assignment. The first, is to become familiar with Unix permissions and access control lists, and to understand how those relate to users and groups. Second, to evaluate a contradictory security policy, and make decisions regarding the most important rules to implement. Finally, to understand that security policies that seem reasonable in natural language may not translate well to security mechanisms.

The majority of questions regarding this assignment has been students asking which security policy rules were more important. However, as the point of the assignment is to give students the experience, and confidence, to make judgement decisions, they are left to make their own decisions. (However, part of the assignment requires them to provide their reasoning behind their decisions.)

CC 2001: OS1 - Overview of operating systems, OS7 - Security and protection.

CHAPTER 5. EVALUATION

5.1 Student Feedback

In order to assess the usability and student acceptance of the Xen Worlds environment, surveys were given to 2 computer security classes that performed at least 2 lab assignments in the Xen Worlds environment. While both classes were an introduction to computer security, one class was at the graduate level, while the other was at the undergraduate level.

The surveys consisted of 9 questions, and were given to approximately 50 students, generating 32 responses - 17 from the graduate class, 15 from the undergraduate class. The possible responses to the survey ranged from 1 strongly agree, to 5 strongly disagree. The questions, and summary of the results can be seen in Table 5.1, and the raw responses in Appendix A [AJD09].

As can be seen from the survey summary, student experience with the Xen Worlds environment is favorable, and many students consider it a valuable tool for learning, and reinforcing class material. In fact, looking at the questions regarding learning, the majority of students did agree the environment and assignments assisted in their learning. The breakdown of the number of responses for these questions can be seen in Figure 5.1 and Figure 5.2.

In addition, students found using the environment enjoyable, with 27 of the 32 response agreeing, or strongly agreeing with this statement. The frequency of responses to this question can be seen in Figure 5.3.

Given the relatively weak approval of the menu system and documentation, verbal feedback was solicited from the students in an attempt to determine the underlying issues. These discussions indicated confusion over the state of the VM, (running or suspended), as there was no physical feedback such as a status LED or fan noise to indicate if the VM was running. This

Question	Mean	Standard Deviation
Prior to this class you were experienced with Linux and the command line interface	1.91	1.09
The Xen Worlds menu system was easy to use and understand	1.94	0.95
The documentation for the Xen Worlds environment was adequate	2.00	0.86
The Xen Worlds assignments were clearly written and understandable	1.88	0.71
The responsiveness and usability of the virtual machines was adequate	1.88	0.94
The Xen Worlds assignments helped to learn and understand the class material	1.69	0.74
The Xen Worlds environment was more convenient than a physical lab	1.56	0.88
I enjoyed the Xen Worlds assignments	1.66	0.83
The Xen Worlds labs an environment contributed to your learning	1.75	0.84

Table 5.1 Xen Worlds Survey Results

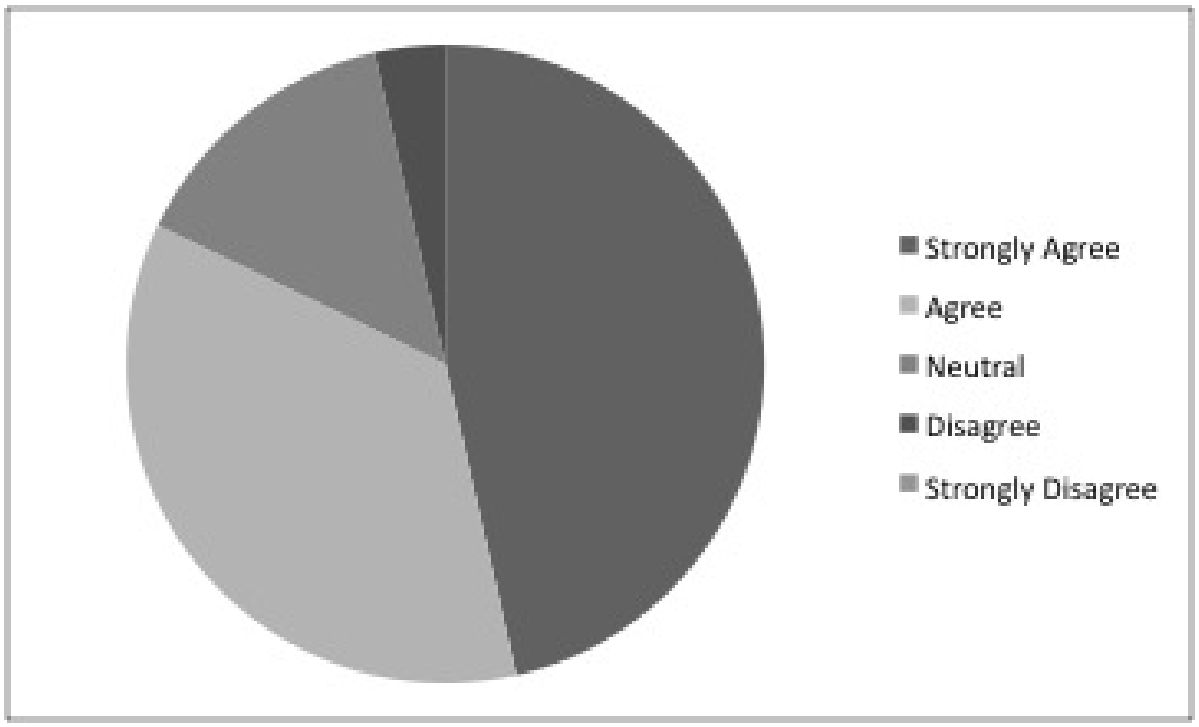

Figure 5.1 The Xen Worlds assignments helped to learn and understand the class material

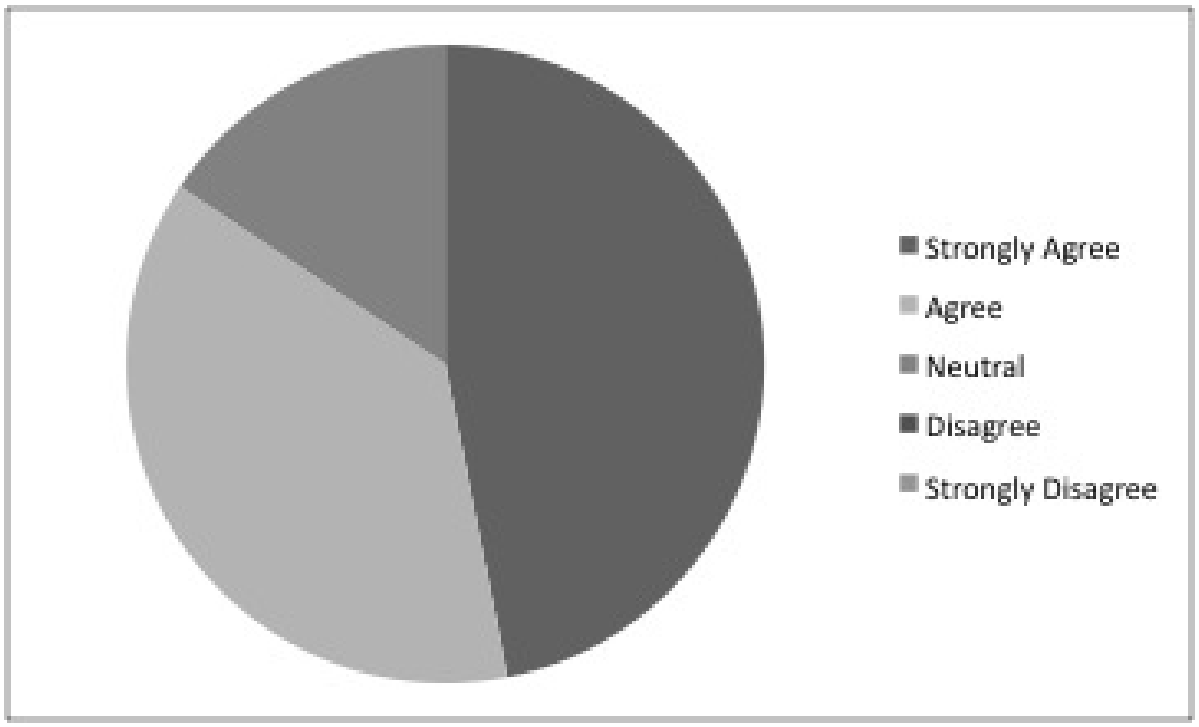

Figure 5.2 The Xen Worlds labs an environment contributed to your learning

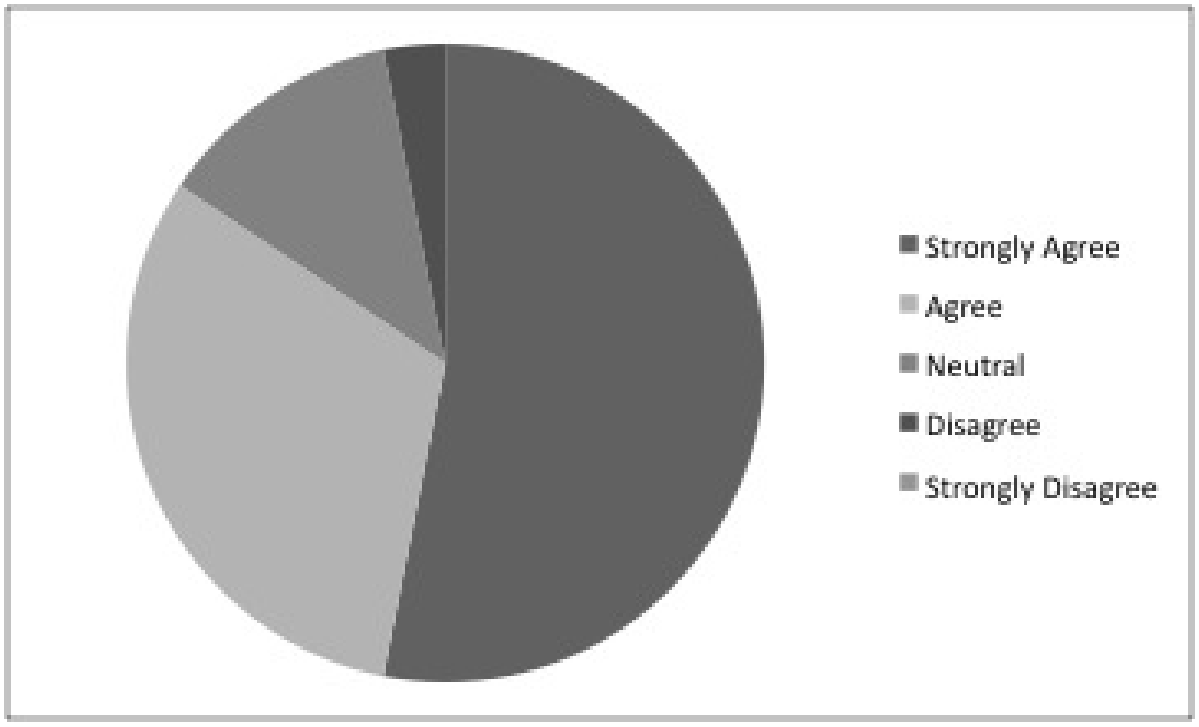

Figure 5.3 I enjoyed the Xen Worlds assignments

lack of feedback from the menu interface often led to frustration on the part of the students.

Given this feedback, the documentation for the Xen Worlds menu was updated to provide a more precise description of the VM status, and alert the students to attempt a start of their VM if they are unable to connect. While no formal analysis has been done, a discussion with the teaching assistant found there were no questions regarding the state of the VM in the most recent class where the Xen Worlds environment was utilized. This class had 35 students utilizing the environment.

Ten of the 32 surveys also included additional comments from the students included additional comments on the Xen Worlds environment, with one student providing two additional comments. The comments can be grouped as follows [AJD09]:

- Six recommended an increase in the number, and complexity, of assignments
- Three indicated issues understanding the assignments, interface or difficulty connecting to the environment
- Two were feature requests for a more robust client, and an IDE available on the VMs.

The 2 comments regarding a more robust environment, do not fit with the usability requirements of Xen Worlds, but are excellent features that are available on other virtual lab environments. The full comments can be found in Appendix A.

Given the overall, positive nature of the feedback from the surveys indicates that the Xen Worlds approach is an effective and enjoyable method of providing lab assignments.

5.2 Performance Analysis

In addition to the survey responses, the Xen Worlds environment was also put through two different performance tests to determine the responsiveness of the environment under load. The first test simulates a number of students concurrently accessing the Xen Worlds environment to simulate a normal use of the environment. The seconds test simulates only a single student accessing the environment, but with the other Xen Worlds running John the

Ripper, (a password cracker), that is a CPU intensive application, to determine the impact of abnormal behavior on other students using the environment.

To simulate student behavior, a number of Expect scripts were created to simulate the general actions students would use when they interact with the environment. The commands included in the script are:

- Read a man page.
- Run a "Hello World" program.
- Run a string-reverse program.
- Switch to a different VM.
- List the contents of a directory.
- Use the find command to locate a file.
- Use the cat command to view /etc/passwd.

Each script executes all of the above commands 3 times. The order of these commands was randomized prior to writing them to the scripts, so the order varies between scripts, but are always executed in the same order by a specific script.

To further simulate student behavior, the actual text of the commands was sent at a rate similar to a human typing, instead of a computer generated burst of text. The specific settings used is:

```
set send_human {.1 .3 1 .05 2}
```

Which, according to the Expect man page, emulates a fast and consistent typist - similar to what we would expect from a student in the computer field.

For both tests, the environment used involves a server running 10 Xen Worlds, with each world consisting of 3 VMs. The scripts are executed on a different machine, and use SSH to connect to the test server.

Number of Simulated Students	Average Execution Time	Standard Deviation
1	107.59	35.44
2	134.71	36.61
3	148.29	30.75
4	147.56	27.31
5	147.61	23.76
6	145.02	24.82
7	149.52	25.25
8	159.20	25.01
9	161.83	25.20
10	171.65	26.15

Table 5.2 Concurrent User Performance Analysis Results

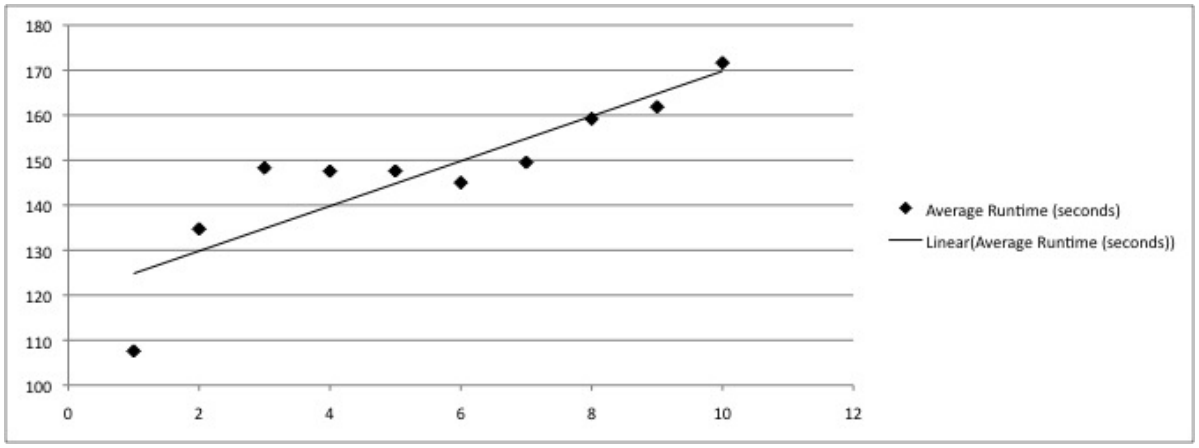

Figure 5.4 Average Runtime: Concurrent User Performance Test

For the concurrent user test, the number of concurrent, simulated students operating within the environment was varied from 1 to 10, with each simulated student operating within their own Xen World. The tests were repeated 10 times for each number of concurrent students. The elapsed clock time for the execution of the Expect script was recorded, with the results displayed in Table 5.2.

Using a linear regression on the data, we calculate a trend line of: $Y = 5.00x + 119.83$, with $R^2 = 0.76$. A graph of the concurrent user performance, and the resulting trend line can be seen in Figure: 5.4.

This data indicates that, with a static number of VMs under interactive load, the system does experience some slowdown, but given the difference in execution times, and the number

of commands executed, it would be only a minor inconvenience to a student. Therefore, we can conclude that with a static number of VMs, under interactive load, the system scales well.

The second test performed is the stress test of the environment. In this set of tests, only a single simulated student accesses the environment, but the number of Xen Worlds that have a VM executing John the Ripper is varied from 1 to 9. This test of the environment is a direct result of verbal feedback received from students regarding sluggish behavior of their VM. The underlying cause was believed to be multiple students running John the Ripper for an extended period of time during the software security lab exercise.

Execution of John the Ripper resulted in a 25% CPU load for the VM executing John the Ripper. Once the number of VMs executing John the Ripper reached 5, this load scaled back, (by Xen), so all of the VMs were utilizing approximately 97% of the CPU resources on the host. The results of the stress test can be found in Table 5.3. However, as seen from the test results, Xen prevents VMs executing CPU intensive applications from starving other VMs on the same host. Figure: 5.5 shows the results with the same scale as the concurrent user performance graph, (Figure: 5.4), and illustrates the minimal impact of CPU intensive applications compared to additional concurrent users.

However, while the high CPU usage does not directly relate to sluggish behavior, it does seem to have a secondary effect on performance. This impact was discovered while capturing screenshots of the Virtual Machine Monitor during testing. When a screenshot was taken, it would take several seconds for the host to scale back the CPU resources for the VMs under load, and provide then to the Dom0, allowing the screenshot to be taken. It then takes several seconds of the Dom0 being idle for those resources to be given back to the VMs.

Given the user script executes continuous operations, it would only suffer a single delay as the VM is provided with the CPU resources needed, but would never be released back due to constant input through SSH. However, since students would not be sending continuous commands to their VMs, each command would encounter this momentary lag of CPU resources being reallocated, and experiencing the sluggish behavior. Fortunately, simply reminding students to not run CPU intensive operations for long periods of time and periodic monitoring of

Number of VMs Executing John the Ripper	Average Execution Time	Standard Deviation
1	118.12	3.80
2	117.19	3.15
3	115.64	3.72
4	116.18	3.72
5	118.75	3.46
6	187.97	3.80
7	118.38	4.44
8	119.00	3.75
9	116.82	3.00

Table 5.3 Stress Test Performance Analysis Results

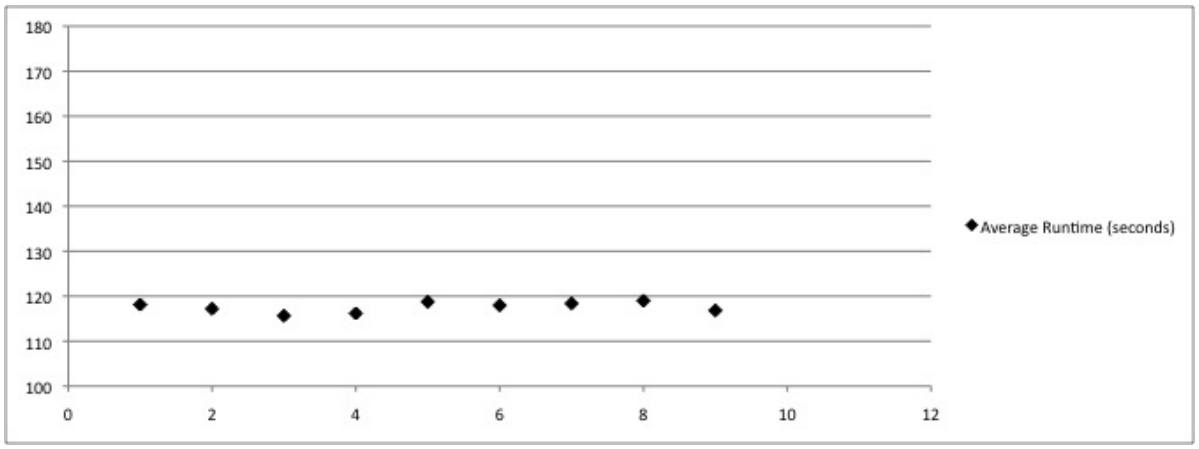

Figure 5.5 Average Runtime: Concurrent CPU Intensive Applications

VMs that have high CPU usage should eliminate the sluggish behavior.

CHAPTER 6. CONCLUSIONS AND FUTURE DIRECTIONS

This thesis has discussed the Xen Worlds environment and its use in various courses at Iowa State University. Xen Worlds does not provide many of the advanced features of other virtualization projects such as a graphical interface, dynamic scaling or student configurable networks, and should not be considered a one-size-fits-all virtual lab environment. However, it does provide an environment that is extremely easy to learn and use, that can run on inexpensive hardware, with low bandwidth and does not require students to have anything beyond a computer and network access.

The assignments created for Xen Worlds cover a variety of curriculum areas, and feedback indicates they are an enjoyable and effective method of teaching, and reinforcing lecture topics. In fact, while few additional comments were received, the majority indicated a desire for a greater number of assignments, illustrating student acceptance of the environment.

The Xen Worlds environment can be provided using minimal hardware, and open source software, making it a low-cost option. Given the ability of the Xen Worlds environment to support several hundred VMs, on a few thousand dollars of equipment, we feel that this approach is a cost-effective way of providing a secure lab environment that is equally accessible to on-campus and off-campus students.

Looking forward, there are several areas of Xen Worlds that could be improved or modified. First, the assignment library could be expanded to include additional assignments, including more complex and open-ended assignments to increase the ways a student can explore the VM environment. In addition, the current assignments can be rewritten into smaller modules allowing instructors to control how they are combined, and have greater control the pace at which they are assigned.

Second, the current limit to providing VMs is caused by a software configuration issue within Xen itself, that limits the number of block devices that can be automatically named. This limit can be configured in some versions of the Xen software, and would allow up to 255 VMs to be run on each server. However, this fix requires installing Xen form source and manually editing the source files, so would violate the requirement for ease-of-use by the instructors.

Finally, since the different assignments have varying configurations, files and applications running, each VM image is specific to a given lab assignment. However, given that Expect is used to configure the VM images, it is possible to load all necessary files on a single VM image and for an assignment-aware script to be executed, installing all required features, and deleting extraneous artifacts. This approach would allow all assignments to be provided on a single 1.5 Gigabyte VM image.

APPENDIX
SURVEY RESPONSES

Questions

The raw responses to the survey questions are in the tables below. For each of the following 9 questions, students were asked to respond according to the scale:

1 - Strongly agree

2- Agree

3 - Neutral

4 - Disagree

5 - Strongly Disagree

A table entry of ẍïndicates students did not answer that question.

Q1: Prior to this class our were experienced with Linux and the command line interface

Q2: The Xen Worlds menu system was easy to use and understand

Q3: The documentation for the Xen Worlds environment was adequate

Q4: The Xen Worlds assignments were clearly written and understandable

Q5: The responsiveness and usability of the virtual machines was adequate

Q6: The Xen Worlds assignments helped to learn and understand the class material

Q7: The Xen Worlds environment was more convenient than a physical lab

Q8: I enjoyed the Xen Worlds assignments

Q9: The Xen Worlds labs an environment contributed to your learning

Q1.	Q2.	Q3.	Q4.	Q5.	Q6.	Q7.	Q8.	Q9.
1	1	2	2	2	1	3	2	2
1	2	2	3	2	1	1	2	1
2	1	1	1	2	1	1	1	x
1	1	3	2	2	2	1	2	2
2	1	1	2	1	1	1	1	1
1	1	1	1	1	1	1	1	1
4	3	3	4	2	2	3	2	2
2	1	2	1	1	2	1	2	1
5	3	x	1	2	1	1	1	1
1	1	1	2	1	1	4	4	x
2	2	2	2	2	2	3	2	3
2	3	3	3	3	3	2	3	3
3	2	3	2	2	2	2	2	2
1	1	3	2	3	2	1	1	2
4	4	2	2	4	1	1	1	2
4	2	3	1	1	1	3	1	1
1	3	4	2	4	2	1	1	2

Table A.1 Graduate Responses

Q1.	Q2.	Q3.	Q4.	Q5.	Q6.	Q7.	Q8.	Q9.
1	2	1	3	2	2	1	3	4
1	5	3	2	4	3	1	1	1
1	2	1	1	1	2	1	1	2
2	2	2	2	1	3	2	2	2
1	2	2	2	1	3	1	1	3
3	2	2	2	2	1	1	1	1
2	2	2	2	2	2	2	2	2
2	2	2	2	2	1	1	1	1
2	2	2	2	3	2	1	2	x
2	2	1	1	1	1	2	1	1
1	1	1	2	1	2	1	1	1
1	2	3	2	2	3	3	3	3
2	2	2	1	1	1	1	1	1
2	1	1	1	1	1	1	1	x
1	1	1	2	1	1	1	3	1

Table A.2 Undergraduate Responses

Additional Comments

How did the Xen Worlds labs and environment contribute to your learning?

- More understading of Linux and security problems.
- The virtual environment, giving root priviliges is very helpful in learning the public key systems, the vulnerability of the ssh ??? From .host files and the assigment are file permissions is very nice.
- The Xen Worlds labs help me learn the real technologies which can be used by hackers.
- I was totally unaquainted with the UNIX commands. Now I know many of them.
- I learnt how to setup access controls for users directories. I get to know network structure for a company/department.
- Liked the access control and privileges.
- Practice.
- Hands on experience is a better learning tool for me.
- Informative, hands-on approach.
- Having full reign of a unix machine is VERY NICE. It prevents dumb overhead. (Not letting into ?? Areas, etc.).
- It is the only Linux environment I have access as a root user. The fact tht the machine is stand alone and not share with other users in learning about the machine.
- Hand on makes security real. Theory is great but does nothing for future security administrators.
- A better understanding of priviledges, how exploits can be used.
- Allowing me to play around with concepts learned in class like access control lists.

- Sneaking around was fun, but I already knew Unix permissions.
- It helped with hands on experience without the risk.
- It showed how the networks and linux system were actually put together.
- hands on is always better than lecture.
- Made me practice some of the concepts we did in class and understand them better.
- Allowed me root access on a system to perform things in Linux I couldn't at a regular lab.
- It allowed whole networks to be traversed without the physical requirement and also allowed us to have root privileges in an environment where in a lab we couldn't do that.
- had I done all the Xen Worlds labs, this survey might be more meaningful.
- Practical implementation of concepts.
- Hands on experience. Covered stuff discussed in class.

Recommendations for improving the Xen Worlds environment or assignments

- Provide client application to access xen worlds environment.
- The assignment should describe more clear
- 1. Provide up-to-date edit and programming environments e.g. python, emacs and related add-on libs. 2. Add some programming assignments.
- More assignments would be helpful for more clear understanding of the unix environment.
- Assignment were really good. If you can increase the number of assignments then it would be better for students.
- If we can set up the environment such that we can conduct real cracking techniques that'll be awesome!

- Ok helped a little.
- Have more of them. Maybe more, smaller assignments.
- Too simplistic. When you say "there is an IDS and an alarm will go off if you do something wrong" mean it. Implement something.
- I really didn't understand the "create", "destroy", etc. portion of Xen Worlds. This sometimes caused problems for me (not getting into the virtual machines) Perhaps better direction/documentation on this would be helpful.
- Reliability. I had multiple instances where connectivity would drop on machine would lose routes.
- Spend more time on advanced topics, setup firewalls, etc. Actual breaking IDS stuff.
- If the window resize issue could be fixed, that would be nice.
- I sometimes had problems logging into my machine, but that may have not been a Xen Worlds Problem.
- Find a fix for the visualizaion problems occuring if terminals are different widths/heights.
- More of them. They were fun to work with.

Any other comments regarding the above questions, or any other Xen Worlds topic

- It is nearly perfect environment. Thank you!
- Very nice tool. The assignment on finding the "Black Pearl project". It was not very clear at the beginning whether we should use the cracking techniques or not. So it took me a long time to examine those cracking methods.
- vhost one was a little difficult to find instructions - found actual items first.

- #7: Depends on how a physical lab makes available to offcampus. #1: Some rusted unix experience ¡empty set¿ Linux experience. #2 and #3: I did not use the menu nor the documentation because of the display it shown on my screen. Regarding man pages, I either look up on spock machine or on the Internet.
- They were fun assignments. Should have been more of them.
- #4: Except last one.

BIBLIOGRAPHY

[AB04] John Aycock and Ken Barker. Creating a secure computer virus laboratory (case study). In *EICAR 2004 Confference CD-rom: Best Paper Proceedings*, 2004.

[AD06] Benjamin Anderson and Thomas E. Daniels. Xen worlds: Xen and the art of computer engineering education. In *Proceedings of 2006 ASEE Annual Conference and Exposition*, 2006.

[AFG+09] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.

[AGR07a] William D. Armitage, Alessio Gaspar, and Matthew Rideout. Remotely accessible sandboxed environment with application to a laboratory course in networking. In *SIGITE '07: Proceedings of the 8th ACM SIGITE conference on Information technology education*, pages 83–90, New York, NY, USA, 2007. ACM.

[AGR07b] William D. Armitage, Alessio Gaspar, and Matthew Rideout. A UML and MLN based approach to implementing a networking laboratory on a scalable linux cluster. *J. Comput. Small Coll.*, 23(2):112–119, 2007.

[AJD09] Benjamin Anderson, Amy Joines, and Thomas Daniels. Xen worlds: Leveraging virtualization in distance education. In *14th Annual Conference on Innovation and Technology in Computer Science Education*, 2009.

[And09] Benjamin Anderson. Xen worlds project. `http://home.eng.iastate.edu/~hawklan/xw-index.html`, 2009.

[BDF+03] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 164–177, New York, NY, USA, 2003. ACM.

[Bel07] Fabrice Bellard. Qemu internals documentation. Wiki: QEMU.org, 2007.

[Cor08] Intel Corporation. Intel virtualization technology for directed i/o. Technical report, Intel Inc., September 2008.

[Des88] Rene Descartes. Meditations on first philosophy. In *Descartes: Selected Philosophical Writings*. Cambridge University Press, 1988.

[Dev07] Advanced Micro Devices. *AMD64 Architecture Programmer's Manual, Volume 2: System Programming*, September 2007.

[Dik00] Jeff Dike. A user-mode port of the linux kernel. In *ALS'00: Proceedings of the 4th annual Linux Showcase & Conference*, pages 7–7, Berkeley, CA, USA, 2000. USENIX Association.

[DTW07] Wenliang Du, Zhouxuan Teng, and Ronghua Wang. SEED: a suite of instructional laboratories for computer SEcurity EDucation. In *SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education*, pages 486–490, New York, NY, USA, 2007. ACM.

[DW08] Wenliang Du and Ronghua Wang. Seed: A suite of instructional laboratories for computer security education. *J. Educ. Resour. Comput.*, 8(1):1–24, 2008.

[EBJ09] Nate Evans, Benjamin Blakely, and Doug Jacobson. A security capstone course: An innovative practical approach to distance education. In *ASEE/IEEE Frontiers in Education Conference*, 2009.

[Hes08] Joey Hess. Pdmenu: Simple to use menu program. `http://www.kitenet.net/programs/pdmenu/`, 2008.

[hKW00] Poul henning Kamp and Robert N. M. Watson. Jails: Confining the omnipotent root. In *In Proc. 2nd Intl. SANE Conference*, 2000.

[KSR+05] K. Krishna, W. Sun, P. Rana, T. Li, and R. Sekar. V-netlab: a cost-effective platform to support course projects in computer security. In *CISSE '05: Proceedings of the 9th Annual Colloquium for Information Systems Security Education*, 2005.

[LR93] E.S. Lee and D.R. Raymond. Menu driven systems. In Allen Kent and James G William, editors, *Encyclopedia of Microcomputers, Volume 11*, pages 101 – 127. Marcel Dekker Inc., New York, NY, 1993.

[LS06] Edith A. Lawson and William Stackpole. Does a virtual networking laboratory result in similar student achievement and satisfaction? In *SIGITE '06: Proceedings of the 7th conference on Information technology education*, pages 105–114, New York, NY, USA, 2006. ACM.

[Mar90] John Markoff. Computer intruder is put on probation and fined $10,000. `http://www.nytimes.com/1990/05/05/us/computer-intruder-is-put-on-probation-and-fined-10000.html?scp=2\&sq=robert+tappan+morris\&st=nyt`, May 1990.

[MM09] Marianne C. Murphy and Marilyn K. McClelland. My personal computer lab: Operating in the "cloud". In *Information Systems Education Journal*, volume 7, September 2009.

[Nak07] Jun Nakajima. Hybrid virtualization - the next generation of xenlinux. In *Xen Conference Japan 2007*, 2007.

[Pro09] The FreeBSD Documentation Project. *FreeBSD Handbook*. The FreeBSD Foundation, 2009.

[Ros04] Mendel Rosenblum. The reincarnation of virtual machines. *Queue*, 2(5):34–40, 2004.

[SC07] Security and Exchange Commission. Commission guidance regarding managements report on internal control over financial reporting under section 13(a) or 15(d) of the securities exchange act of 1934, June 2007.

[Ser09] Amazon Web Services. Amazon elastic compute cloud (amazon ec2). `http://aws.amazon.com/ec2/`, 2009.

[Sin04] Amit Singh. An introduction to virtualization. `http://www.kernelthread.com/publications/virtualization/`, January 2004.

[SJSM06] Stuart Schechter, Jaeyeon Jung, Will Stockwell, and Cynthia McLain. Inoculating SSH Against Address Harvesting. In *The 13th Annual Network and Distributed System Security Symposium*, San Diego, CA, February 2006.

[SKKS08] Weiqing Sun, Varun Katta, Kumar Krishna, and R. Sekar. V-netlab: an approach for realizing logically isolated networks for security experiments. In *CSET'08: Proceedings of the conference on Cyber security experimentation and test*, pages 1–6, Berkeley, CA, USA, 2008. USENIX Association.

[VMw07] Inc. VMware. Understanding full virtualization, paravirtualization, and hardware assist. Technical report, VMware, Inc., 2007.

[VMw09] VMware. Reduce it costs with a virtualization plan. `http://www.vmware.com/solutions/consolidation/`, 2009.

[Woo07] Anita Woolfolk. *Educational Psychology*. Pearson Education, Inc., 2007.