2006-2300: XEN WORLDS: XEN AND THE ART OF COMPUTER ENGINEERING EDUCATION

Benjamin Anderson, Iowa State University

Thomas Daniels, Iowa State University

Dr. Thomas E. Daniels is an Assistant Professor in the Department of Electrical and Computer Engineering at Iowa State University in Ames, Iowa. Tom received his Doctorate in Computer Science from Purdue University under the advisement of Eugene H. Spafford. He did his graduate work at Purdue, initially in the Computer Operations, Audit, and Security Technology (COAST) Lab and then in the Center for Education and Research in Information Assurance and Security (CERIAS).

Xen Worlds:
Xen and the Art of Computer Engineering Education

Abstract

Xen Worlds is being developed to provide a method for performing assignments and lab work in information assurance, operating systems and networking courses that require root access to the individual machines, or the entire network. Currently, there is no existing approach that addresses the root access requirement and the entire life-cycle of an assignment from problem definition, to turn-in of the end product. The Xen Worlds project is aimed at creating a versatile "virtual lab" where an entire network of virtual machines, (a Xen World), can be provided to each student that will allow on-campus and distance education students 24/7 access via SSH, allow students to turn-in a virtual machine or an entire network as the finished product and allow for grading to occur directly on those machines instead of grading a few select artifacts such as configuration files, programs or outputs.

To achieve these goals, Xen Worlds builds on several open source products, including utilities for creating virtual network bridges and the Xen virtual machine monitor developed at the University of Cambridge. Combining these building blocks and custom software into a versatile architecture, it has been possible to create an environment that supports multiple backends; provides administrative tools to the students for starting, stopping, resetting, saving and turning-in their virtual machines; isolating the Xen Worlds from each other; and allowing the instructor direct access to the virtual machines for grading purposes.

The potential of this project was demonstrated by creating a prototype that provided 30 students with their own virtual machine for an entire semester, with all 30 virtual machines being run on a single desktop computer with a Pentium 4 processor and 2GB of RAM. An upper limit of 30 virtual machines being provided was dictated by the amount of physical RAM required by each virtual machine on the desktop computer. The current version of Xen Worlds spreads the virtual machines over many backend computers, allowing up to 240 virtual machines in arbitrary network topologies. Our current goal is to prove Xen Worlds' potential as an instructional tool, and demonstrate its lower cost compared to commercial solutions such as VMWare. A milestone towards this goal is to provide 1000 virtual machines continuously operating for an entire semester to 200 students in a variety of classes using less than $10,000 in commodity hardware.

Introduction

Practical experience through lab work has long been recognized as an important part of an engineering education. Familiarity with various software systems and equipment is critical to the future success of computer science and engineering students. However, in some areas such as operating systems, networking and information assurance, the ability to access the desired functionality requires administrator, or root, level access to the individual machines or an entire network. This level of access can lead to security and privacy issues with the systems used, since students are literally able to manipulate and modify the systems in any way they desire.

Past efforts have used many different ways of handling these issues, such as having labs dedicated to specific courses[1], or using virtualization software such as VMware[2] or User-mode Linux (UML)[3] to provide a virtual machine. However, there are no existing approaches that address the issues of scalability, usability (for both on-campus and distance education students) and security, while also addressing all of the requirements and usability in all phases of the assignment life-cycle.

The Xen Worlds project is creating a versatile "virtual lab" where every student is provided root access to an entire network of virtual machines, (a Xen World), created for their use in the course. This is accomplished through the use of open source products, including the Xen virtual machine monitor developed at the University of Cambridge, and custom software developed for the Xen Worlds project. The consoles for the virtual machines (VMs) are accessed through the use of SSH, providing the same interface to both on-campus and distance students, reducing the complexity of having both on-campus and distance students within the same class. In addition, Xen Worlds allows for the turn-in of an entire virtual network instead of a few select artifacts such as configuration files, programs, outputs or screen shots. This allows for grading to occur directly on the VMs as the instructor or teaching assistants are able to run the VM to evaluate its behavior. Finally, Xen Worlds can achieve these goals with a relatively modest hardware cost and no software cost.

The prototype of the Xen Worlds project was introduced in the senior-level course CprE 431X, Basics of Information Security, in the Electrical and Computer Engineering Department at Iowa State University during the Spring 2005 semester, and has been greatly expanded for the next offering of the course in Spring 2006. This paper will discuss the architecture of the Xen Worlds prototype and the assignments that were given to the students using this prototype. It will also present the new architecture and interface that has been adopted, and the motivations behind these changes. Once the architecture has been presented, we will examine how Xen Worlds addresses the requirements and phases of the assignment life-cycle and analyze the performance of the system. This paper will close with a discussion of the future development plans for the Xen Worlds project.

Xen Prototype: Architecture and Assignments

The potential of the Xen Worlds project was demonstrated with the implementation of a prototype server that hosted VMs for 30 students for an entire semester. The hardware requirement for this prototype was modest, and it was implemented on a standard desktop system that was already in use by the teaching assistant assigned to the course and continued to be used as their personal system in addition to being the Xen Worlds server. The system used was a commercial desktop with a Pentium 4 processor and 2 GB of RAM with 100 GB of hard drive space. The operating system used was Red Hat Enterprise Linux Workstation v.4 with Xen 2.0 installed. Access to the VMs was provided through a stepping stone method where students would SSH into the Xen Worlds server machine and then request a console to their VM. Since Xen requires root access to handle the VMs, the Xen tools were accessed through wrapper scripts to prevent students from accessing VMs other than their own. An upper bound of 30 virtual machines existed due to the physical RAM required by each virtual machine.

Xen uses a unique approach to virtualization in that an OS needs to be ported to the Xen architecture, and run above the Xen layer. For the OS used in the prototype, this was accomplished simply by installing the Xen package and recompiling the kernel; no modification of code was required. Xen also eliminates many of the traps and interrupts that can slow performance in a virtual machine environment by having Xen run in privilege ring 0, and the OS run in privilege ring 1. Privileged instructions are replaced by hypercalls to the Xen interface which prevents virtual machines from interfering with each other at the hardware layer. This provides a very efficient hypervisor layer, and testing by the Xen development group showed that performance of Xen is near that of the native OS, with a worst-case performance reduction of 8%. The same tests performed with UML or VMware showed performance reductions of up to 88%[4].

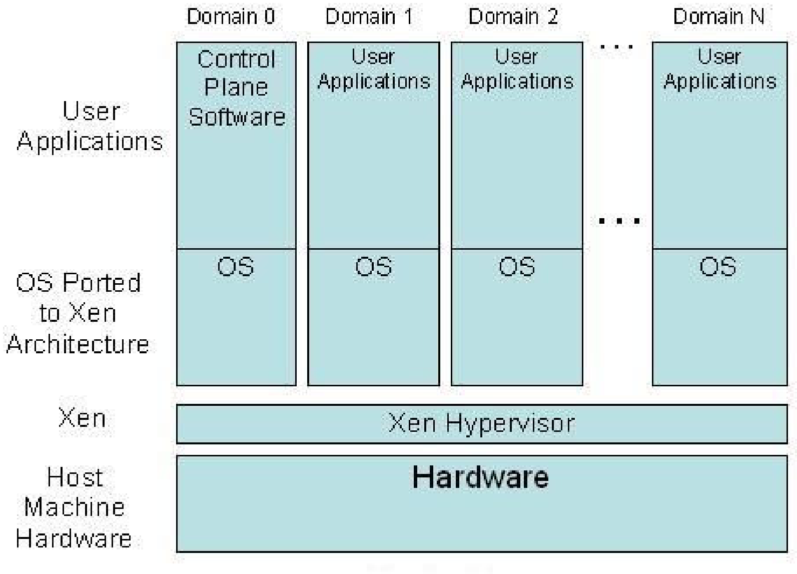


Figure 1: Xen Architecture

In our initial prototype, Xen Worlds allowed for several assignments that would not have been possible without the ability to provide each student with their own VM. The first assignment was to configure the root account to allow authentication through knowledge of the teaching assistant's private key, allowing the teaching assistant to grade assignments on the VM without knowledge of the root password set by the student. Another assignment had the students running malicious tools to execute buffer overflow attacks and escalate privileges using an insecure remote access server. In a traditional lab environment, this assignment could not have been given, since the risk of the system being compromised by an outside attacker would have been too great. The final assignment involved giving the students a theoretical company, including divisions, departments, employee lists, directory structure and security policy. The students then had to create user accounts and groups for the employees and implement the directory structure with permissions set to enforce the given security policy. This assignment allowed the students to apply many of the topics covered throughout the course, and provide practical experience dealing with a real-world situation.

Current Xen Worlds Architecture

Having success with the Xen Worlds prototype, a new architecture was developed that would allow for much greater scalability, and would provide greater usability and security to the students. It was determined that simply increasing the number of servers was not an efficient and scalable solution, as an increase in the number of VMs would require a corresponding increase in the number of machines required, with a corresponding increase in administration overhead. In addition, it was determined that requiring the students to learn the proper Xen command usage and syntax could detract from the overall learning objectives, so a simpler interface was required; one that would require a minimal learning curve. Finally, it was determined that providing students command line access on the Xen Worlds server could be a security risk without a dedicated system administrator maintaining a proper configuration.

The solution for addressing the scalability issue was to use a small cluster of diskless computers that would use a single 1U computer for network booting and system services. The current Xen Worlds configuration consists of 8 diskless computers, each with a Celeron 2.0GHz processor and 2 to 4 GB of RAM, which are simply referred to as "boards" since they lack a hard drive, case, and have their mother boards directly mounted to a small metal rack; and a single 1U computer, also mounted in the rack. The boards are connected to the 1U using Gigabit Ethernet, and the 1U is connected to the campus network with 100Mbit Ethernet. Each board uses the Preboot Execution Environment (PXE), or "pixie boot", to manage booting. PXE uses DHCP to acquire an IP address from the 1U and then perform a network boot, also from the 1U. Once booted, the boards access their files on the 1U by using a network file system (NFS). The images for the VMs are also provided by the 1U, and are mounted as network block devices. By eliminating the overhead costs of hard drives, cases and monitors, and using open source software, the overall cost of the system is greatly reduced. The entire cost of the Xen Worlds cluster is under US$7,000, and using Fedora[8] as the virtual machine OS allows 300 VMs to be run.

To improve the usability and security of Xen Worlds, the command line interface on the Xen Worlds server was replaced with Pdmenu[5], a menu driven shell that restricts the commands available to the students and abstracts the Xen administrative commands. This simplifies the Xen administrative tasks as it eliminates the need to learn the underlying command syntax and hides commands that the students will not require, such as the memory allocation functions. The use of Pdmenu also provides more security as it restricts the commands available to students on the Xen Worlds server.

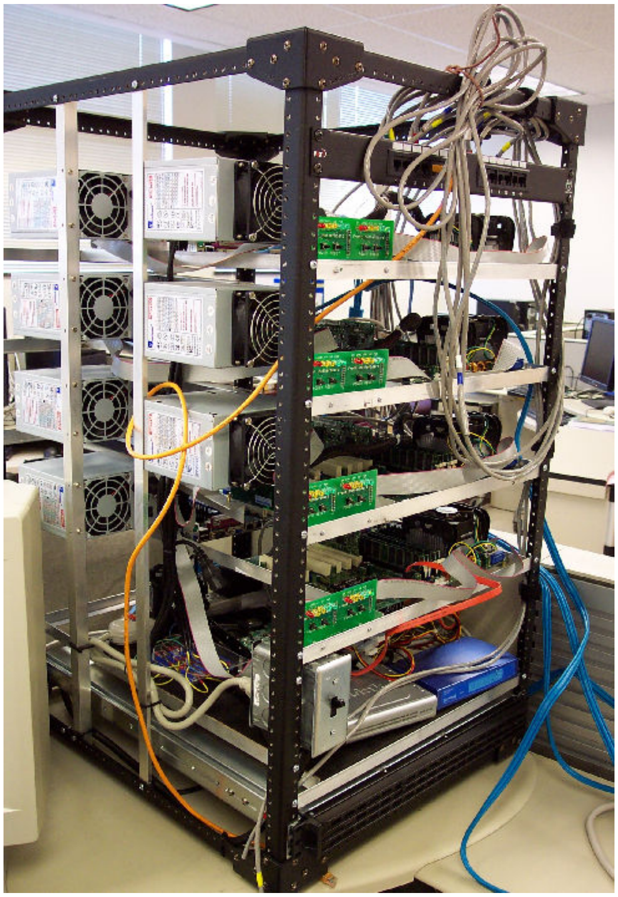
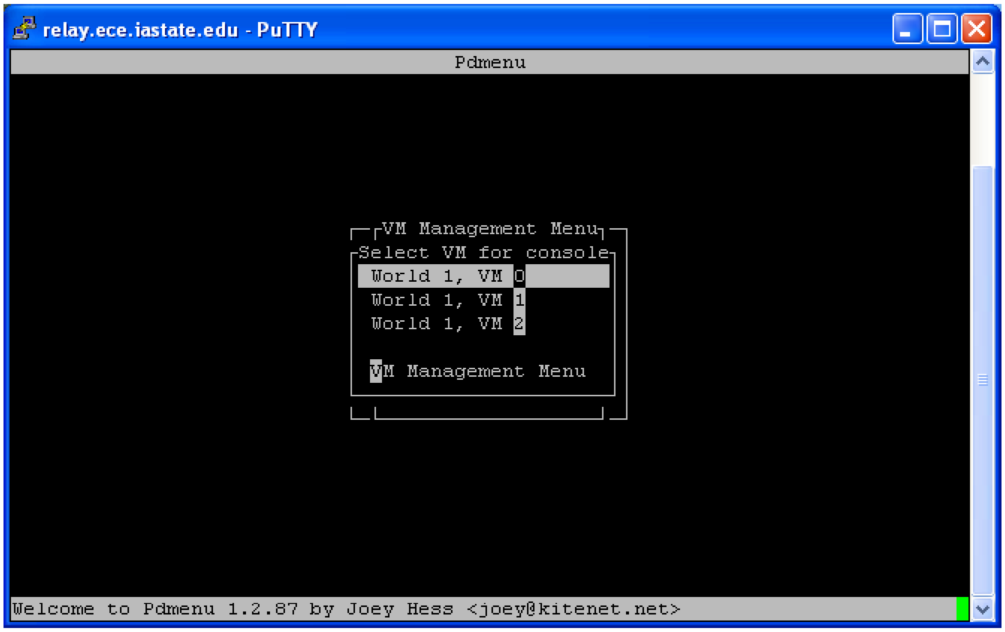


Figure 2: Current Xen Worlds Cluster



```
                                   Pdmenu




                             ┌┌VM Management Menu┐┐
                             ┌Select VM for console┐
                             │ World 1, VM 0       │
                             │ World 1, VM 1       │
                             │ World 1, VM 2       │
                             │                     │
                             │ VM Management Menu  │
                             └─────────────────────┘
                             └└───────────────────┘┘




Welcome to Pdmenu 1.2.87 by Joey Hess <joey@kitenet.net>
```

Figure 3: Pdmenu Interface

Access to the Xen Worlds servers is similar to the prototype, as the students access the server through the use of SSH. However, to make access more transparent, students SSH to a specific TCP port on the 1U system and destination network address translation (DNAT) is used to route the packet to the appropriate board on the cluster's internal network. This access method also protects the boards from outside attackers, since connections to the boards must go through the 1U, and the 1U only forwards packets to the SSH port on the boards. This provides a secure and transparent connection to the back-end of the cluster.

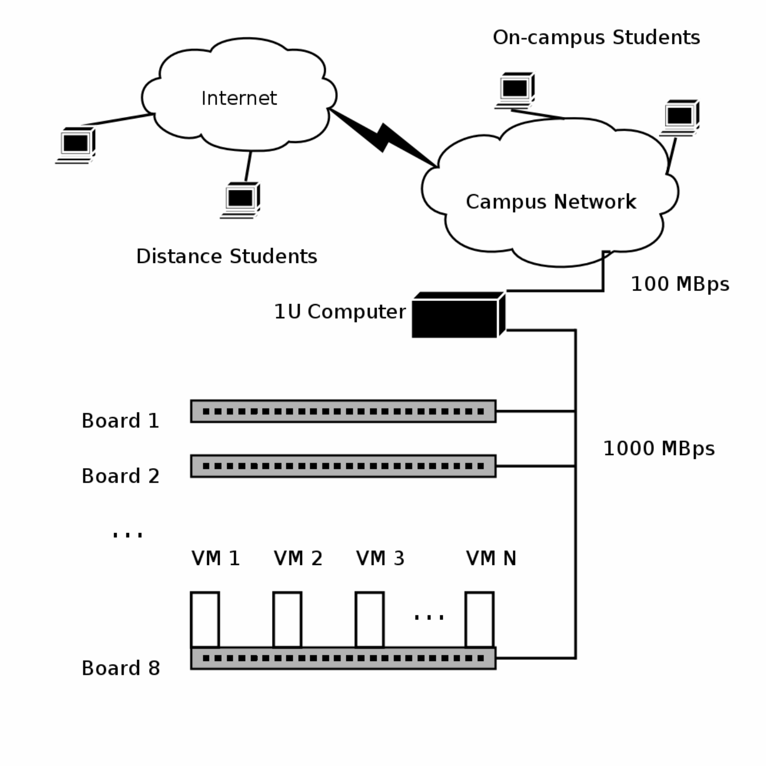


Figure 4: Current Xen Worlds Architecture

Addressing the Assignment Life-cycle

The Xen Worlds project is designed to be used in a teaching environment, and is tailored to the requirements and phases of the assignment life-cycle, ensuring ease-of-use for the instructor and students. In this section we will examine what methods Xen Worlds uses to support and simplify the various requirements and phases of the assignment life-cycle.

Requirements

Uniformity and stability of the development environment: Since the Xen World servers are accessed through the use of an SSH connection, on-campus and distance students are provided with the same interface, eliminating the need to develop two sets of assignments. In addition, since the Xen World servers are abstracted from the students, any changes made to the underlying system are not visible to the students.

Availability of the development environment: Since no physical access to the Xen Worlds server is required, the system can be available 24/7, and can be accessed from any system with an Internet connection. This is particularly important when there is no formal lab scheduled and students have to arrange the time to work on the assignments around their academic and personal schedules.

Ability to save or checkpoint progress: While the file system of the VM itself can be used to save the work done by the students, Xen also provides administrative tools where an entire VM can be saved to a configuration file and then restored at a later time. This allows students to save their VMs before making potentially hazardous changes.

Life-cycle phases

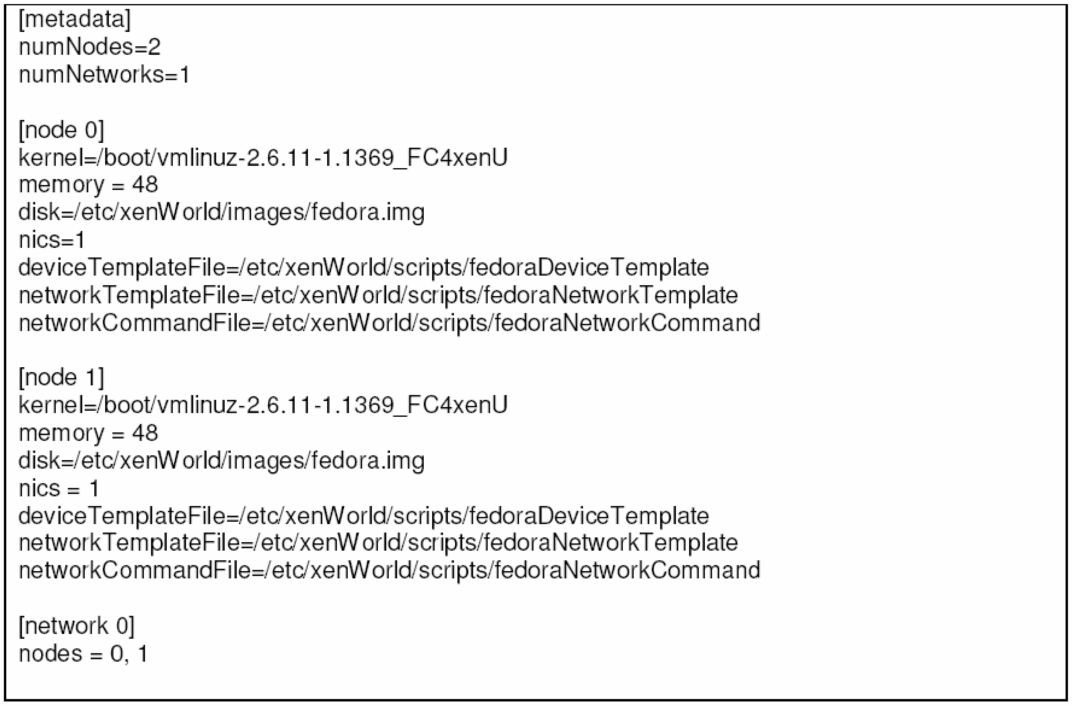
Assignment design: Xen Worlds provides tools where an instructor can define the layout of a Xen World in a single configuration file, including: the number of nodes, the configuration of those nodes, network topology, node naming and addressing, and account creation. See figure 5 for a sample configuration file creating a Xen World with 2 nodes, attached to a single network.

Assignment Implementation: Once the proper configuration has been determined, all of the desired Xen Worlds can be generated with a single command and includes copying of all required files, assignment of IP addresses and subnetting, creation of all required virtual network bridges and startup of all required services.

Availability/handout of the assignment: Since the Xen Worlds are accessed through the use of Pdmenu, it is a simple matter of including the appropriate menu entries when the assignment is released. The menu entries are generated from a plain text configuration file, so a simple copy of the configuration file at the appropriate time is required, and the new menu entries will be available on the student's next login. Also, since the Xen World naming scheme is based off the username of a student, only the student logged into a particular username can access the associated Xen World, preventing other students or outsiders from interfering with a student's work.

Assignment turn-in: As mentioned previously, Xen provides the functionality for the state of a VM to be saved to a configuration file that can be loaded at a future time. For turn-in, a student can use this functionality to save their entire world and have the resulting files copied to a turn-in directory, where it can be accessed by the instructor or teaching assistant.

Grading and return of assignment: Since the Xen World is saved using the save functionality built into Xen itself, it is a simple matter for the world to be restored and accessed by the instructor or teaching assistant. Grading can also be done on another dedicated system separate from the cluster. Once grading is completed, the world can be saved and made available for the student to examine. It is also possible to provide a copy of a "solution world" to the students that they can restore and examine.



```
[metadata]
numNodes=2
numNetworks=1

[node 0]
kernel=/boot/vmlinuz-2.6.11-1.1369_FC4xenU
memory = 48
disk=/etc/xenWorld/images/fedora.img
nics=1
deviceTemplateFile=/etc/xenWorld/scripts/fedoraDeviceTemplate
networkTemplateFile=/etc/xenWorld/scripts/fedoraNetworkTemplate
networkCommandFile=/etc/xenWorld/scripts/fedoraNetworkCommand

[node 1]
kernel=/boot/vmlinuz-2.6.11-1.1369_FC4xenU
memory = 48
disk=/etc/xenWorld/images/fedora.img
nics = 1
deviceTemplateFile=/etc/xenWorld/scripts/fedoraDeviceTemplate
networkTemplateFile=/etc/xenWorld/scripts/fedoraNetworkTemplate
networkCommandFile=/etc/xenWorld/scripts/fedoraNetworkCommand

[network 0]
nodes = 0, 1
```

Figure 5: Xen World Configuration File

Performance Analysis

To determine the performance of the current Xen Worlds architecture, multiple expect scripts were created to simulate how a user would interact with the system. The scripts connected to the Xen Worlds through an SSH connection and executed 24 commands that could be performed by a standard user. The behavior simulated included editing files, find commands, execution of programs and switching VM consoles. These expect scripts were executed on a separate workstation and the real clock time required for the execution of the expect scripts on that separate workstation was measured.

The Xen Worlds environment used for the analysis involved a single board within the cluster that was running 10 Xen Worlds, with 3 VMs in each world, to simulate providing a small network to 10 students. To examine the change in performance under different load conditions, the number of simulated students accessing the VMs at a single time was varied from 1 to 10, with each student accessing their own Xen World. The simulations were repeated 10 times at each load condition.

The results of the experiment are displayed in the table below, with the slowest execution times being with 4 users, and a minimal slowdown of approximately 10% from a single script running to 10 concurrent scripts running. Since Xen provides a CPU time slice to each VM in a round-robin manner, we assume that this is the overwhelming factor for the consistency in execution times. In future work, the analysis will be performed with only the Xen Worlds being tested active. By doing so, it can be determined if the limiting factor is the user input or the processing time of the VMs.

Table 1: Performance Analysis Results

Number of Scripts Running Concurrently	Mean Time to Complete Scripts (s)	Standard Deviation
1	116.95	4.45
2	119.34	16.51
3	119.06	14.66
4	132.74	37.51
5	128.04	26.18
6	124.73	24.52
7	130.43	27.78
8	128.55	23.72
9	126.70	19.22
10	130.75	17.02

Analysis of Mean Run Times

Although a cursory look at the data above suggests little degradation of performance a user load increases, there is a subtle trend in the mean run times. Using a simple linear regression with 95% confidence, we find that runtime increases by 1.29 s for every added user in this range. Hence, for a static number of virtual machines under interactive load on a given board, the Xen system scales well.

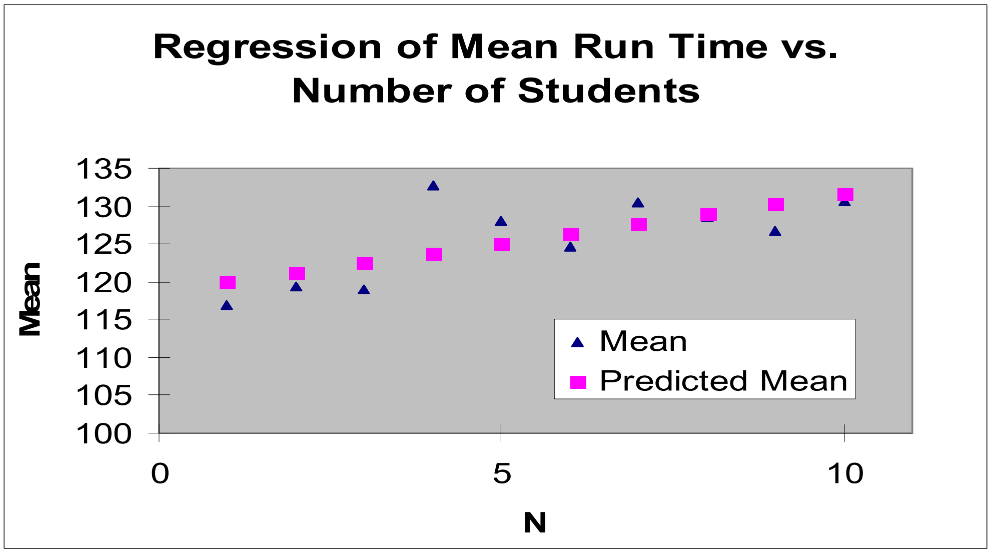


Figure 6: Comparison of Mean Run-time to Linear Regression

With the current performance of the Xen Worlds cluster, and the amount of physical RAM provided to each VM, if each of the boards within the cluster is provided 2GB of RAM, the cluster will support 240 VMs.

Future Directions

The target milestone for the Xen Worlds project is to provide 1000 VMs for 200 students over an entire semester with only a small increase in the RAM provided to each board. It is anticipated that the larger number of students will cause other areas of the system to become the performance bottleneck, so several different methods to improve performance are being examined.

The first method under examination is to add a hard drive to one of the boards and have it act as the network block device server for the cluster. While this will increase the load on the specific board, it will reduce the load on the 1U, which also has to handle the students' SSH traffic and providing services to the boards themselves.

The second method is to modify the priority of the VMs that are running. If there are no students logged into a Xen World, then the scheduling priority of those VMs can be reduced, increasing the CPU cycles available to the active VMs.

Third, porting smaller Linux distributions to the Xen environment may reduce the amount of RAM and storage space required for each VM. The overall goal is to create a VM that will only require 8MB of RAM and 100MB of disk space, and use these VMs as virtual routers so larger and more complex Xen Worlds can be provided.

Finally, it is hoped that different methods for disk management such as a union file system[6] or a Copy-on-Write (CoW) NFS[7] can be used to eliminate the redundant portions of the VM images. Currently, each VM running a Fedora server image is given 2GB of disk space; however, using a union file system or CoW NFS, it should be possible to reduce this to a total of 2GB plus 100MB per VM. In an environment with 50 VMs, this would result in the storage requirements dropping from 100GB to 7GB.

Conclusions

While the Xen Worlds project is still in its early stages, each stage has illustrated the great potential for this approach. In a single generation of the project, the number of virtual machines that can be provided has been increased by an order of magnitude, while the creation and management of the virtual worlds has been simplified. With the rapid progress being made, it is felt that in the near future the Xen World project will reach the proper level of maturity where it can be released as open source software, and be simple enough to be used almost immediately by someone with only a general familiarity with Linux or Xen.

Acknowledgements

The authors would like to thank the Iowa State University Information Assurance Center, Dr. Doug Jacobson, and the ISEAGE project for providing the hardware used to build the Xen Worlds cluster.

Bibliography

1. R. Chapman, W. Carlisle. “A Linux-based Lab for Operating Systems and Network Courses” *Linux Journal*. 01-Sept-1997. <http://www.linuxjournal.com/article/2361>.
2. D. Ragsdale; D. Welch; R. Dodge “Information Assurance the West Point Way”. *IEEE Security & Privacy*. Vol. 1, Iss. 5. 2003, pp. 64-67.
3. G. Heatly “Implementation and Evaluation of a Virtual Computing Environment created with User-mode Linux”. *Graduate School of the Faculty of Design and Technology Conference*. University of Central Lancashire, Nov-2004. <http://www.uclan.ac.uk/facs/destech/gradschool/conference/dec2004/Heatley-Guy.pdf>.
4. “Xen Performance”. University of Cambridge Computer Laboratory. 5-Nov-2004. <http://www.cl.cam.ac.uk/Research/SRG/netos/xen/performance.html>.
5. J. Hess. “Pdmenu: Simple to use menu program”. <http://www.kitenet.net/programs/pdmenu/>.
6. “A Stackable Unification File System”. File systems and Storage Lab, Stony Brook University. 15-Jan-2006 <http://www.fsl.cs.sunysb.edu/project-unionfs.html>.
7. R. Ross. “Copy-on-write NFS server”. 5-Jan-2006. <http://www.russross.com/CoWNFS.html>.
8. T. Chung. “Fedora Project”. 17-Jan-2006<http://fedoraproject.org/wiki/>.