In the Zone: Virtual Computing on a Budget
Stan J. Thomas
Wake Forest University
Winston-Salem, NC 27106
sjt@wfu.edu
Paul M. Whitener
Wake Forest University
Winston-Salem, NC 27106
whitenpm@wfu.edu

ABSTRACT
In this paper, we report on our efforts, extending over several years, to provide computer science students experience with a variety of operating system and computing environments. We describe our explorations into the use of virtual machine environments for instructional purposes, explorations that have led to the current multifaceted approach to virtualization. We also demonstrate that implementing a diverse, sophisticated virtual computing environment does not require a large investment in computer hardware, in fact it can lead to a cost saving by extending the useful life of systems and reducing the complexity of system administration.

Categories and Subject Descriptors
K.3.2 [Computers and Education]: Computer and Information Science Education; D.4.0 [Software]: Operating Systems-General

General Terms
Management, Economics, Experimentation

Keywords
Virtualization, operating systems, computing environments

1. INTRODUCTION
Currently, many colleges and universities struggle to provide students with the latest technologies for learning while working within their budgetary constraints. Computer Science departments, in particular, seek to provide experience with the latest in computing hardware and software for students, whether it be personal computers, mobile devices, high performance computing, or computing in the cloud. However, reality may be a little different. With limited budgets and limited system administration support, programs may believe there is no option but to standardize on one hardware platform and one operating system. This paper reports on our efforts over a five year period to provide a variety of computing experiences for computer science students, culminating in our current multifaceted approach which takes advantage of virtualization tools running on commodity hardware to provide a variety of computing environments at a very reasonable cost. Within this environment we provide students a variety of computing experiences that prepare them for initial employment as well as life-long learning.

2. UBIQUITOUS COMPUTING
For more than a decade Wake Forest University has been a leader in the campus wide laptop computing initiative in higher education [2,3,5]. The university was the second in the nation to offer a common notebook platform to incoming students and to then replace it with a newer model after two years. This comprehensive plan, initially referred to as the Plan for the Class of 2000 [1], led to the elimination of traditional computer labs, including those labs where computer science students previously had access to SGI and Sun Microsystems workstations, IBM personal computers, and other hardware and software platforms. With the elimination of computing labs, the Computer Science department was forced to identify new approaches to provide multiple computing environments to our students.

Subsequently, the department became an early adopter of dual booting student laptops with both Linux and Microsoft Windows. The dual boot approach allowed students to have access to the application rich Windows software load provided by the university—necessary for general class work across campus—in addition to Red Hat Linux—needed for certain computer science coursework. Using the Linux load, students had root access on their own notebooks and could make modifications at the system level. In conjunction with the formal study of operating systems, students gained the ability to “look under the hood” of their own computers. Other classes, beginning with the CS2 course, utilized the Linux load and open source tools for software development in C++. Faculty and support staff from the campus Information Systems group worked to identify a rich set of Linux applications that computer science students could use both for coursework and their own entertainment needs. Students were provided both Linux and Windows updates through university satellite servers and had full access to the software repositories maintained by the campus Information Systems staff.

As time passed there were recurring, hard to diagnose disk instability issues with the dual boot configuration. Knowledgeable students started replacing the university provided Red Hat Linux with other dialects of Linux. Issues arose. Configurations became unsupportable by the campus Help Desk. Faculty and department staff members were frequently asked by students to troubleshoot laptop issues unrelated to the assignment or learning objectives at hand. The dual boot configuration was deemed by the department to be unsustainable. After four years of offering a dual boot laptop load to computer science majors, the decision was made to discontinue providing this option beginning with the Fall 2008 semester. We elected to return to the earlier days when students received their laptops with only the university standard Microsoft Windows load. A new option had to be identified to provide students with experience working in a variety of computing environments in a more engaging way than simply logging into a remote server.
3. VIRTUALIZATION

During the 2007-08 academic year some of our classes were taught using the Linux-like Cygwin environment under Windows. Although Cygwin met some of our needs, it did not provide the full Linux experience desired and it was not an ideal environment for teaching operating system internals. In addition, much like with the dual boot approach, use of Cygwin on student laptops required substantial student cooperation in the installation and maintenance of the software.

During the same time period, three converging developments led the department to explore the use of virtual computing (VC) to provide a variety of computing environments for our students. First, as previously described, the Windows-Linux dual boot option, as well as Cygwin, had proven unsatisfactory. Second, faculty members had very specific computing needs for courses they were introducing, some of which required students to have full system administration or root privileges. Third, the department was fortunate to have hired a system administrator with extensive experience managing virtual computing environments and who was eager to use the latest software tools to support our instructional needs.

We investigated successful virtual computing initiatives in higher education, particularly the Virtual Computing Lab (VCL) project at North Carolina State University [4,6] in partnership with IBM, but we were not equipped nor budgeted to build our instructional activities on a centralized high performance computing foundation. As a small department in a small university we did have the flexibility of experimenting with leading edge technologies in a manner appropriate to our local needs and budget; hence we started our virtualization initiatives by targeting two courses with very specific computing needs.

3.1 Initial Experiments

For the Spring semester of 2008 one faculty member was planning a server administration course utilizing AMP (Apache, MySQL, PHP) tools. For this course each student needed a dedicated computer with root privileges. In years prior to 2008 we used a collection of older desktop computers for these student learning platforms. This approach had proven to be very time-consuming for the instructors, as systems had to be initially imaged and frequently reimaged during the semester. In addition, these weakly secured systems tended to be attractive to hackers.

For Spring 2008, a second faculty member had very specific software requirements for a senior/graduate level computer graphics course he was teaching. He wanted students to develop applications using OpenGL libraries in a Linux environment. This requirement could have been met by utilizing a single Linux/Solaris server but it was feared that the computational requirements for the class would tax a single server.

3.1.1 Platform

At the time, the department had several Sun Blade 1000 workstations in storage. These circa 2000 workstations had originally served as office systems for faculty. In order to support our virtualization experiments, six Sun Blade 1000 systems were salvaged and put back into service. Each unit was configured with 1GB of ram, a single 32GB hard drive and a single 750MHz Sparc III processor. OpenSolaris (Solaris Express Community Edition) was installed as the operating system (OS). This OS release was chosen primarily because of its support of Solaris Zones and secondarily because of the wide variety of open source tools bundled with it.

Solaris Zones have some unique features that make them extremely easy for administrators to maintain. A Solaris Zone is a kernel level virtualization technique comparable to a BSD Jail. It is a lightweight method that provides complete, isolated, and secure runtime environments for users and applications with negligible resource overhead. Any number of individual non-global zones (virtual machines) can be allocated, secured and managed by an instance of Solaris called the global zone. Physical hardware resources and kernel operations are managed by the global zone and allocated dynamically to its hosted non-global zones. In the configuration we employed, non-global zones shared common file system directories as read only mount points. This reduced the needed disk footprint per zone to around 150Mb. Each non-global zone can have its own personality, including separate password/shadow files, hostname, and IP addresses, and may be given access to shared or dedicated resources at varying priority levels. Application software packages may be added to individual non-global zones as needed. By design, the non-global zone memory space and disk “user land” are isolated via the kernel from the global zone as well as from other hosted non-global zones. This makes them very secure and allows independent rebooting and “crashing” of the non-global zone without affecting other zones on the box. Each non-global zone is an ideal environment for student work requiring “virtual” root access. At the same time, the environment helps to consolidate common Solaris administration tasks to a small number of Solaris global zone instances.

A non-global zone was designed for the AMP class. Once the configuration was approved by the instructor, the zone was easily replicated with a distinct password/shadow file, hostname, and IP address for each. One zone was created for each student in the class for a total of twelve AMP zones across six Sun Blade 1000s. Each student thus had root access to the equivalent of a full Solaris server but in a virtualized environment.

Similarly, a non-global zone was configured for the graphics class containing the desired development environment and graphics libraries. Once the “load” was approved by the professor, one zone was created for each student in the graphics class for a total of twelve graphics instruction zones across the six Sun Blade 1000s.

Students in these classes used a variety of methods to connect to their virtual machine zones. Students needing only a command line interface simply used putty under Windows or ssh in a terminal session from other operating systems. Students needing a graphical interface from Mac OS X or Linux used a terminal window to run ssh and X-Windows provided the graphical interface. Students running Windows on their client computers had access to both HummingBird Exceed and the open source combination of xming with putty. A small number of Sun Ray thin clients were also provided for students to access their zones when their personal computer was not available.

While students in the AMP class were not required to utilize their personal zone, their other option was to provide their own hardware, which some students elected to do. Similarly, while students in the graphics course were not required to use their zone for class, if they did choose to use their zone, they could be
assured that the professor had tested all assignments and verified operation on the zones. This made debugging and troubleshooting much easier for both student and professor. In addition, students from both classes had full access to the zones, including root access. The zones were accessible from both on campus as well as off. Students could reboot their zone at their own will, make changes and administer their own server, all while not affecting other zones on the host or the host itself.

3.1.2 Lessons Learned

Once they were configured and created, the only problem with the zones supporting AMP administration had to do with permissions that had to be modified across all the zones. The problem arose from an oversight in the configuration of the load and not the system itself. The hardware, software platform was stable, performed well, and no substantive issues were reported.

For the graphics course, a few students were hampered by an incompatibility between the version of the OpenGL Utility Toolkit (GLUT) running on their zone and the version running on their own Linux devices. On these systems, some GLUT routines did not interact correctly between the zone and the graphics hardware on the client machines. The issue was not observed on client systems running OpenSolaris or Mac OS X. Again, the computing performance and reliability of the Sun Blade 1000s was adequate, although the network traffic between client and server was, at times, a noticeable bottleneck.

From a system administrator’s point of view, managing Solaris Zones is straightforward. From the global zone, the administrator can shut down a non-global zone, reboot it, and change any password without affecting other zones. Software patching is also assured 24 virtual servers to be created across six otherwise shelved Sun Blade 1000 workstations, as reflected in Table 1. This resulted in a monetary savings for the department as well as an easily configured, easily managed, and stable work environment for students and the faculty. Students were also able to have their own servers with root privileges, an added benefit.

### 3.2 Year Two

Based on our successful foray into virtualization during the Spring 2008 semester, the department elected to move additional courses to a virtual computing environment for Fall 2008. To prepare for increased dependence on virtualization the department purchased two multiprocessor, multicore servers. These SunFire x4600 servers contained eight dual core AMD64 2.8GHz CPUs with 32GB of ram and dual 146GB hard drives.

In planning for the Fall 2008 semester the faculty identified a need for four different operating system environments in addition to Windows XP: OpenSolaris, Debian Linux, Ubuntu Linux, and a Windows server (to support SQL Server). Although a single Linux/Solaris environment would have been sufficient, we elected to support the specific requests of individual faculty members. For example, Solaris was desired for a course needing the ultimate hardware performance, whereas Ubuntu Linux was desired for an OS course in which students need to study and modify kernel source code. These varying requirements led to a multi-faceted solution.

3.2.1 Multiple Platforms

Once again, the Sun Blade 1000s were a part of the solution. They were reconfigured to provide zones for students in three courses. Three different zone configurations were deployed to support software development in our data structures course; a SAMP (Solaris, Apache, MySQL, PHP) environment for a database management course; and a basic configuration for a class learning Perl and Postgres. The SAMP environment was supported on a single shared zone to facilitate discussion of multiuser database concepts. In all other cases, students had their own dedicated zones.

A second component of the solution utilized our new SunFire x64/x86 servers. These servers were configured with OpenSolaris which offers an additional virtualization tool named xVM specifically for x64/x86 processors. xVM is Sun’s release of Xen, the open source virtualization platform. xVM/Xen can be viewed as an open source equivalent to VMware Server and is classified as a Type 1 hypervisor. A Type 1 hypervisor runs on the “bare metal” of the server. The guest VM is made to think it is the only OS running. Sun’s implementation of xVM/Xen offered the benefit of being part of the OpenSolaris release, not an add on package. One of the SunFire servers was configured to double as both an xVM virtualization host and to provide native OpenSolaris for a second level course on data structures and algorithms. Using OpenSolaris with xVM, fully virtualized instances of Debian Linux were created on the new servers. These instances were used for an upper division/graduate level operating system class, as well as a graduate level networking class. Students had root access to their own virtual machines and were responsible for administering them. xVM also hosted an instance of Windows running Microsoft SQL Server for a database management class.

The final piece of our virtualization solution for Fall 2008 arose from a need to provide a Linux-on-the-laptop solution without dual booting but while retaining the extensive Windows software load provided by the university. The answer was VirtualBox, another virtualization tool that is free for personal and academic use. Originally developed by the German company Innotek, it is now distributed by Sun in conjunction with the xVM initiative. VirtualBox is a Type 2 hypervisor product much like VMware Server and Workstation. That is, it runs on top of an OS as opposed to being part of the OS. VirtualBox is available for almost all x64/x86 based systems, including Windows, multiple Linux versions, OpenSolaris, and Mac OS X. Once Virtual Box

### Table 1. Virtual Computing Platforms – Spring 2008

<table>
<thead>
<tr>
<th>Course</th>
<th>Computing Need</th>
<th>VC Solution</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMP Administration</td>
<td>AMP with root access</td>
<td>OpenSolaris Zones</td>
<td>12</td>
</tr>
<tr>
<td>Graphics</td>
<td>C++, GLUT, and libraries</td>
<td>OpenSolaris Zones</td>
<td>12</td>
</tr>
</tbody>
</table>
is running, users can then install virtualized installations of Windows, most versions of Linux, or the x64/x86 Solaris strains. The installation and setup of a VM under VirtualBox is simple. In fact, the installation of VirtualBox, as well as the downloading, installation and setup of Ubuntu under VirtualBox served as Lab 1 for the students in our CS2 course. We have even used VirtualBox successfully in a Digital Media course with a mixture of CS majors and non-majors. Once installed, Ubuntu runs as a guest OS inside a “window” on the students’ Windows standard load, providing a standard GUI interface to Ubuntu within the window. This configuration also allows students to move files between Windows and Ubuntu via a common VFAT folder.

3.2.2 Lessons Learned

As we experienced in Spring 2008, there were very few problems or issues with our use of Solaris Zones in the fall semester, even though they were running on legacy hardware.

Our first experience with the xVM architecture on the SunFire platform was very positive in terms of the capability of the hardware to support a large number of simultaneous virtual machines. We did have an issue with a few of the VMs supporting Debian Linux as the guest OS. These VMs would occasionally go into a sleep state that required them to be “rebooted” from the host server. This condition was random and difficult to reproduce. In fact the exact cause of the problem was never identified. Instances of Ubuntu Linux running on the same system at the same time never exhibited this problem behavior.

The VirtualBox software proved to be remarkably stable and was quickly endorsed by students. We found it to be a very effective replacement for the traditional dual boot configuration. The most significant problem that arose during the semester was caused by students accepting automatic updates to the Ubuntu kernel in the guest OS. In a few cases these kernel updates required reinstallation of the VirtualBox guest addition tools, even though they never caused problems with Windows itself or to the VirtualBox software itself.

During the Fall 2008 semester, the combined use of these three approaches to virtualization created almost 90 virtual machines running on various platforms, including student laptops and legacy workstations. Once again, this clearly resulted in a monetary savings for the department as well as an easily legacy workstations. Once again, this clearly resulted in a running on various platforms, including student laptops and approaches to virtualization created almost 90 virtual machines during the Fall 2008 semester, the combined use of these three approaches to virtualization created almost 90 virtual machines running on various platforms, including student laptops and legacy workstations. Once again, this clearly resulted in a monetary savings for the department as well as an easily managed, and stable work environment for students and faculty.

### Table 2. Virtual Computing Platforms – Fall 2008

<table>
<thead>
<tr>
<th>Course</th>
<th>Computing Need</th>
<th>VC Solution</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to Linux</td>
<td>Debian Linux</td>
<td>xVM</td>
<td>8</td>
</tr>
<tr>
<td>Linux Administration</td>
<td>Debian Linux w/root</td>
<td>xVM</td>
<td>6</td>
</tr>
<tr>
<td>Perl&amp;Postgres</td>
<td>Root privileges</td>
<td>OpenSolaris Zones</td>
<td>3</td>
</tr>
<tr>
<td>CS2 (C++)</td>
<td>Ubuntu</td>
<td>VirtualBox</td>
<td>12</td>
</tr>
<tr>
<td>Data Structures</td>
<td>C++ compiler, libraries</td>
<td>OpenSolaris Zones</td>
<td>8</td>
</tr>
</tbody>
</table>

### Table 3. Virtual Computing Platforms – Spring 2009

<table>
<thead>
<tr>
<th>Course</th>
<th>Computing Need</th>
<th>VC Solution</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Server Administration and PHP</td>
<td>Root privileges</td>
<td>OpenSolaris Zones</td>
<td>6</td>
</tr>
<tr>
<td>CS2 (C++)</td>
<td>Ubuntu</td>
<td>VirtualBox</td>
<td>12</td>
</tr>
<tr>
<td>Programming Languages</td>
<td>Multiple Compilers</td>
<td>OpenSolaris Zones</td>
<td>12</td>
</tr>
<tr>
<td>Operating Systems</td>
<td>Debian Linux</td>
<td>xVM</td>
<td>22</td>
</tr>
<tr>
<td>Data Mining</td>
<td>Standard Login</td>
<td>OpenSolaris Zones</td>
<td>8</td>
</tr>
</tbody>
</table>

3.3 Third Semester

By Spring 2009 we had started to rely very heavily on virtualization for our instructional computing needs. The system administrator developed a streamlined process for configuring and deploying virtual machines. Otherwise, the Spring 2009 semester proved to be a repeat of the Fall 2008 semester. The distribution of virtual machines utilized for the semester is shown in Table 3.

### Table 3. Virtual Computing Platforms – Spring 2009

<table>
<thead>
<tr>
<th>Course</th>
<th>Computing Need</th>
<th>VC Solution</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Server Administration and PHP</td>
<td>Root privileges</td>
<td>OpenSolaris Zones</td>
<td>6</td>
</tr>
<tr>
<td>CS2 (C++)</td>
<td>Ubuntu</td>
<td>VirtualBox</td>
<td>12</td>
</tr>
<tr>
<td>Programming Languages</td>
<td>Multiple Compilers</td>
<td>OpenSolaris Zones</td>
<td>12</td>
</tr>
<tr>
<td>Operating Systems</td>
<td>Debian Linux</td>
<td>xVM</td>
<td>22</td>
</tr>
<tr>
<td>Data Mining</td>
<td>Standard Login</td>
<td>OpenSolaris Zones</td>
<td>8</td>
</tr>
</tbody>
</table>

4. CURRENT DIRECTIONS

At the current time (Fall 2009) almost all of the instructional computing in the department is supported partially or completely on virtual machines. We are now conducting a pilot using Sun's Virtual Desktop Infrastructure (VDI) server. The product is similar to the North Carolina State Virtual Computing Lab (VCL) platform without the scheduler function [4, 6]. Using either Sun's implementation of VirtualBox or VMware, VDI allows virtual machines to be built and “served up” on a demand basis. So far, VDI has proven to be very flexible. The VDI software allows virtual machines to be made available for one time use and then destroyed or for them to be assigned for an arbitrary length of time. Virtual machines can be built on a Windows system running the special version of VirtualBox that ships with VDI, or they can be built natively on a Solaris server with VDI installed. Any x86 operating system can be loaded on the virtual machines, including Windows, Linux, and Solaris. (Mac OS X is only licensed to run on Apple hardware.) Sun's VDI server has a wide choice of retention and provisioning options that make the system well suited for class use. The initial VDI software install took about an hour and the system was up and running with a working
Ubuntu virtual machine in another 2 hours. The VDI server software only runs on Solaris, but can serve out almost any x86 platform, except for OS X, as mentioned. Students can connect while on campus or from home using the VPN connection provided. As an extension of VDI, Sun's Secure Global Desktop (SGD) software allows students to connect to the VDI server through a web interface from anywhere in the world. We are studying the purchase of licenses for the SGD software but have not yet made the decision to invest in this technology.

Our experience with the use of virtualization for instructional purposes has been very positive. Virtualization has allowed us to provide many different operating systems and environments for faculty and staff with modest investments in new hardware, while putting back into use older, unused hardware. It has allowed us to do more with fewer servers, and hence to reduce overall energy costs. And it has helped to familiarize our students with the types of computing environments they are likely to work in as they enter today's work force.

5. Acknowledgments
The authors would like to thank the faculty members who have contributed to this learning process by employing virtual computing environments in their courses: Dr. Jennifer Burg, Dr. Daniel Canias, Dr. David John, Dr. Tim Miller, Dr. Paul Pauca, Dr. William Turkett.

6. References