AC 2009-2305: TEACHING WEB DEPLOYMENT WITH OS-VIRTUALIZATION

Michael Bailey, Brigham Young University

Joseph Ekstrom, Brigham Young University

Teaching Web Deployment with OS-virtualization

Abstract

While hardware-level virtualization systems such as VMware are widely used in academia, the use of operating system virtualization offers benefits of scalability that are far greater. Since 2004 Brigham Young University has provided an operating system level virtual machine (VM) to each student in the introductory web systems course of the IT program. Each VM forms a capable web server platform, with Apache and MySQL providing dynamic HTML capabilities. This has had many benefits to student learning, forcing students to become familiar with remote access to hardware via command-line, file transfer, and other similar concepts that are common to commercial web development.

Since the virtualization system used allows good performance with over 200 lightly loaded virtual servers on a single hardware server, we have allowed students to retain their virtual servers throughout their academic careers, and to use them to serve their own non-commercial web pages. This has promoted additional independent and service learning opportunities for students. Students have hosted web sites for nonprofit or church organizations, family groups, and have developed portfolio materials to show to future employers.

Introduction

Early in the development of the IT program at Brigham Young University we created a course that incorporated an introduction to the core concepts of Information Technology from the perspective of web development and deployment[2,3]. As we designed the course we were confronted with the problem of providing a cost-effective deployment experience for each student. We did not have budget or space for multiple computers per student so we began to explore virtualization as an alternative. We had been using VMware but found that we could only run 2-3 LAMP (Linux, Apache, MySQL and PHP/Perl) servers on our available hardware and were thus still confronted with smaller but insurmountable budget, hardware, and space barriers. Since our goal was to deploy about 100 identical LAMP servers, we didn't need the flexibility of a full Virtual Machine Monitor that emulated the hardware. We turned to a more scalable solution that we discovered was being used by a local web hosting company: OS-Virtualization using Virtuozzo by SW-Soft (now Parallels).

In this paper we will first give a brief history of the use of the term "virtual machine" from the early 70's until the present. We explain the different models that have been implemented under the term and how usage of the terms seems to have stabilized recently. We then look at one approach in greater detail, OS-Virtualization (sometimes called OS containers), or creating a virtual operating system interface rather that a virtual hardware interface.

Virtualization Methods

The topic of Virtual Machines and especially server virtualization has received extensive attention recently.[5, 10,11,12,13,14,14] A particularly good introduction to virtualization was provided

by M. Tim Jones in December 2006.[4] He points out that the term "virtual machine" has been used at least since the late 1960's to describe providing the illusion of access to a complete computer even though the resources are being shared.[6] It was applied to some early time-sharing systems and was also delivered as a named product as "VM", an IBM 360 Model 67 supervisor which was created specifically to allow software to run as if it were on the older hardware without change and thus facilitate IBM's ability to up-sell their customer base to 370 and even to it's current Z9 (which can run 1970's OS360 software) on 40 year newer equipment. Since typical terminology of the times called the OS the "Supervisor" by the mid 70's the virtual machine manager (VMM) was being called the "hypervisor".

In contrast to the usual resource configuration of a server (Figure 1), where a single operating system manages all the applications and their access to hardware, these VM's gave the operational software the illusion of a complete hardware system's execution environment. This is called called *Hardware Virtualization* (Figure 2). VMware Server and VirtualBox are typical of this type of system. Another approach called *Paravirtualization,* of which Xen is and example, replaces the kernel or drivers of an OS with special versions created to interface with the VMM and avoid the need for interrupts. The necessary specialized low-level code actually contains a significant part of the emulation code so this approach often achieves better performance (Figure 3). And finally one can virtualize at the layer of the OS interface so that the application software calls on a *virtual private server* (VPS) delivered by the *Operating System-level Virtualization* (Figure 5). This approach requires a special 'abstracted' OS kernel but achieves near-native performance.[8] This paper discusses using Virtuozzo and OpenVZ, OS-virtualization systems.

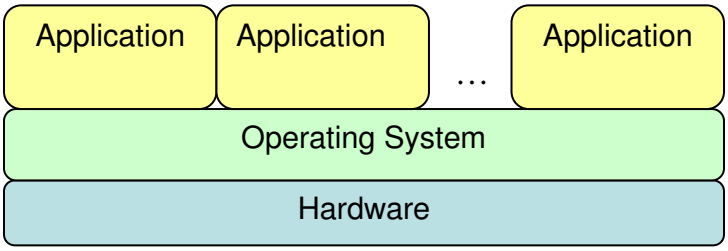

Figure 1. Ordinary server layers

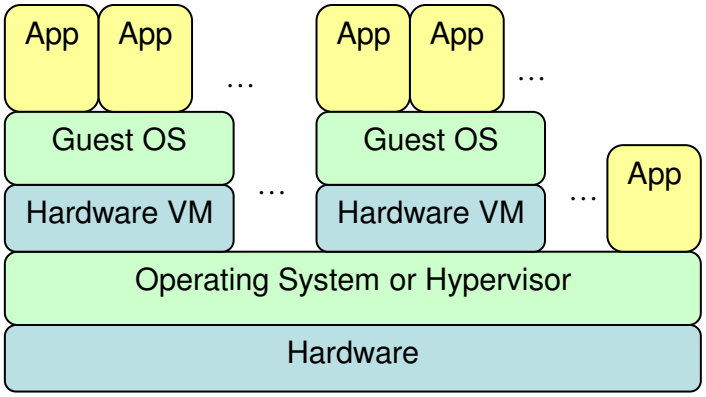

Figure 2. Hardware emulation uses a VM to simulate the required hardware

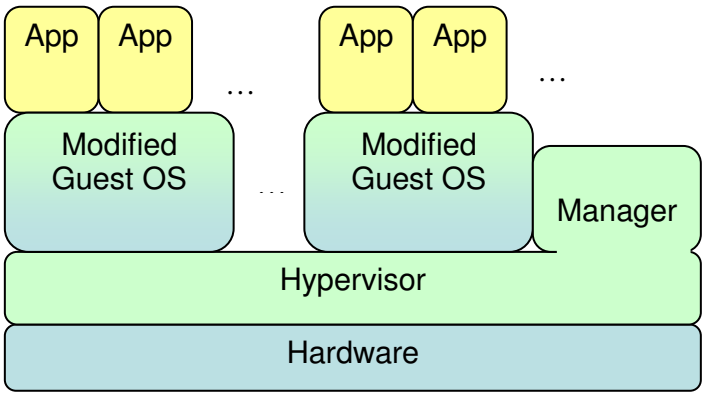

Figure 3. Paravirtualization shares the management with the guest operating systems

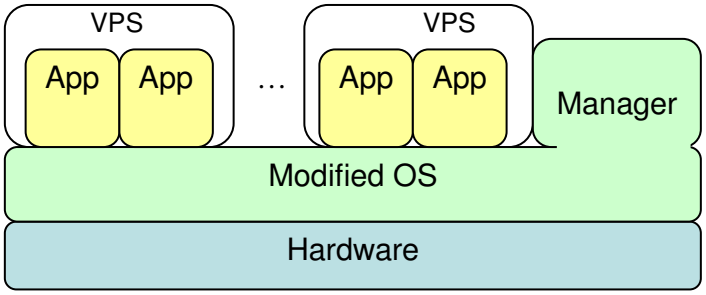

Figure 4. Operating system-level virtualization isolates apparent servers

Other types of virtualization, of less interest to this paper, are included here for completeness. The term virtual machine has also been used to describe the byte-code interpreter used as the portable substrate for running compiled Java code, the Java Virtual Machine (JVM). In this case the "virtual machine" is designed to run a particular language well and might never be implemented in hardware. This Jones calls *Processor Virtualization.* A related approach is what Jones calls *Instruction Set Virtualization.* Instruction Set Virtualization can be implemented as a software emulator, this approach is often used to develop software for a CPU that is under development.

Web Hosting Systems

To better understand the goals of this pedagogy, a brief discussion of web hosting systems is in order. While many web systems are high-volume, dedicated server farms for major corporations, many other web sites are lightly used web presences for small to medium sized businesses. Many of these businesses do not have dedicated IT staff, nor the desire to maintain IT equipment. Such businesses will use the services of a web hosting provider.

Since web hosting is an easily obtained network service can exist at any physical location, it is very cost competitive. With prices starting at $4 per month, it is obvious that the providers are not dedicating a $1000 server to host each web site, rather, some method of resource sharing is necessary. While a shared machine with individual user accounts may suffice for very low

service and security applications, a virtual machine for each user provides much greater assurance.

To maximize profits, the greatest server utilization possible is desired, while providing acceptable response times and reliability. For example, Google sets up its server farms so that they have around 50% utilization on its servers.[1] Higher utilization results in poor performance, and lower is not cost effective. In contrast to Google, where servers are routinely added to reduce over-utilization, the typical small business' single web server is very lightly utilized, frequently only a few percent, on average. As a result, a single server using a low-overhead virtualization technique can host 50 to 300 such web sites.[2] While it is not practical for an introductory web development course to provide a Google-like experience, experience with such a high capacity machine with many virtual private servers (VPS) can be provided.

The Student System

For our sophomore web systems course, a single infrastructure server is used to provide a VPS (or container) to each student. In addition, students are allowed to retain their VPS throughout their academic career. Additional virtualizations are maintained on the server for faculty and student projects. As a result, the server has approximately 200 virtual containers on it.

This number of containers is possible because on average, each is lightly used. In addition, the server incorporates capable hardware and software. The hardware is from a commodity supplier, but includes four Intel XEON processors, 16 GB of RAM, and a 500 GB RAID. The virtualization software used for the first 3 years of its implementation was Parallel Corporations' Virtuozzo, but in the past year this has been replaced with OpenVZ as a cost-cutting measure. Since OpenVZ is an open-source derivative of Virtuozzo, this change required only minor effort.

Each VPS is configured as a LAMP server, with Linux, Apache, MySQL Perl, and useful utilities pre-installed. Students access the servers through Secure Shell (ssh), transferring files to and from the server and using its command prompt to administrate the system. Each VPS is assigned a unique IP address on the laboratory network. For each user on his or her console, the appearance is that he is on his own system, with no other users. The user has a root and user accounts, with most of the capabilities that come with them. Each user can see his VPS' log files and update the configuration files.

There are, of course, some limitations. The most obvious one is that if things go seriously wrong on a VPS, there is no physical machine to manually reboot. A laboratory administrator must be found to reboot the container from a master control utility. On the other hand, as a testament to the quality of the isolation of the containers from each other, we have had no instances in which a student's application has been able to damage a VPS other than his own.

Another limitation from a teaching perspective is that in order to maximize the efficiency of the system, software that is common to all the containers on the server should be pre-installed by the system administrator. This software image is then shared by all the VPS. This is not desirable if we need to give students experience with system administration tasks such as installing and

doing the initial configuration of server software. Many instructors may not consider these as serious limitations.

Course Results

There are 8 laboratory assignments in the semester of this web development course. Over the course of each assignment, students will use a new technology for delivering dynamic web content: Cascading Style Sheets, Javascript, PHP, Perl, MySQL, etc. In order to effectively debug each assignment, development is done on a local machine. Since many of the technologies learned execute on the server, this local machine must also have the necessary web server software installed.

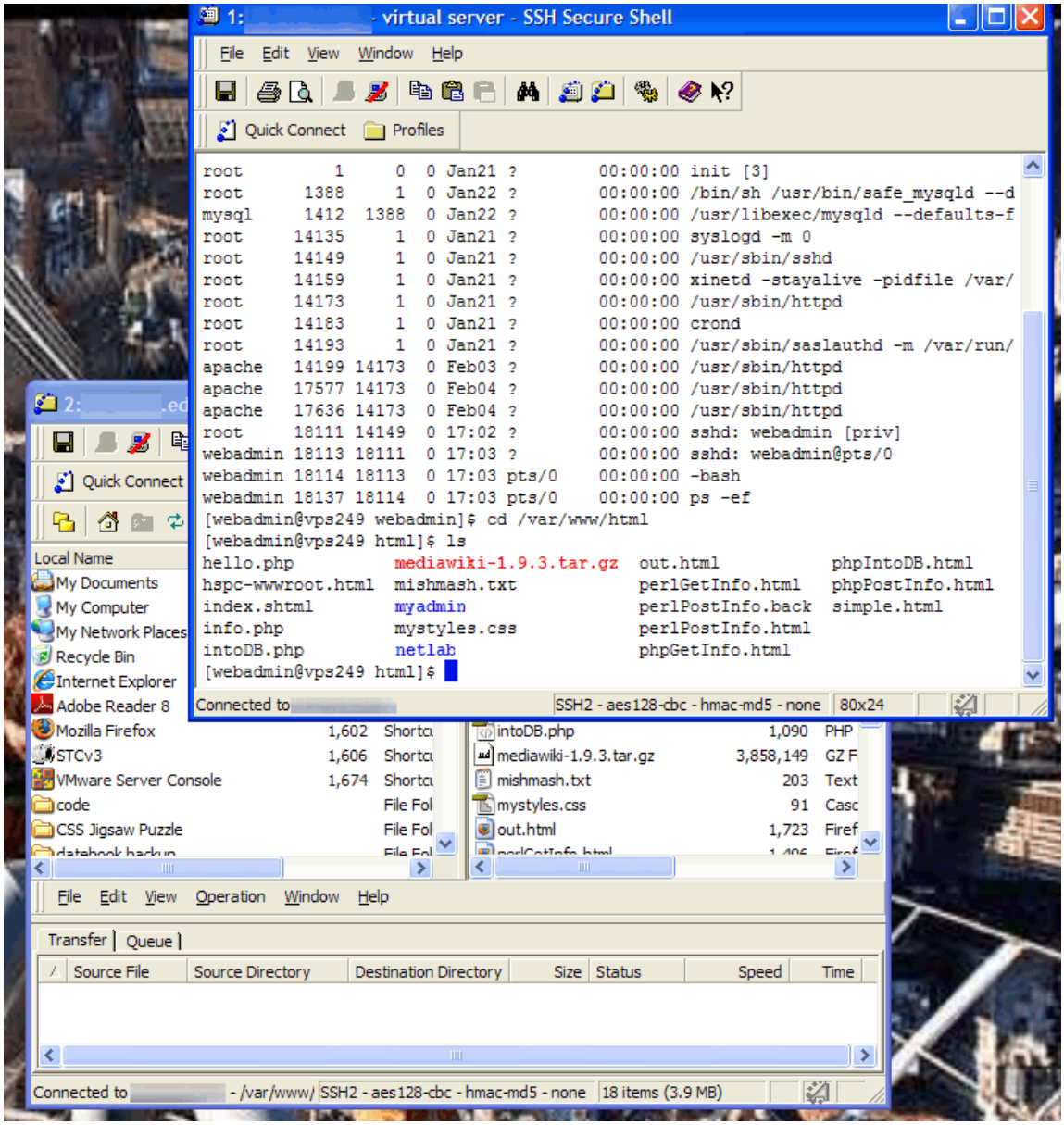

Figure 5 - ssh utility

When the individual components of the assignment have been adequately tested and debugged on the local system, the student initiates an ssh session with his VPS system (Figure 5). The student then transfers files to the web source directory of his VPS. He will frequently need to adjust permissions settings, passwords, and Apache configuration files to achieve the lab goals. Debugging will require the student to access the appropriate log files. If a needed process like MySQL is not operating, the student will need to start it, or better yet, add it to the system boot-up processes.

All of these tasks are good *NIX system administration experiences for beginning IT professionals. Even for IT professionals who are more interested in development than in administration careers, these concepts are vital to understand, since the develop locally, deploy remotely paradigm is prevalent in industry. If not, one could envision a case where a Perl script was put onto a deployment server, and a developer not understanding why its default file permissions prevent it from executing. Additionally, exercises like this that reinforce command line experience are desirable.

Other Benefits

Upon completion of the course, the students are allowed to retain their VPS until graduation, and do any ethical and non-commercial thing that they may desire with it. Many do not use this resource after this sophomore-level course, but others do for other courses or personal projects. In a survey of students who had completed this course, nearly 10% had developed additional web projects on their VPS. Typical of this are two students who developed and hosted a web site for their church's women's auxiliary.

The survey also showed that 45% of students had shown their VPS web sites to friends, family, or potential employers. This is notable not only as a source of pride to the students, but as a strong provider of positive publicity to the program and as a recruiting vector for future students and their employers.

Finally, 48% of the students, including almost all of those who took jobs in web development while in college, went on to develop web sites that were placed on commercial hosting services. The experiences learned in this course, with the assistance of the VPS, prepared them for these opportunities.

Conclusions

While not solely attributable to the VPS lab configuration, this course has been very successful with our IT students. Among other things, it is the most favorably mentioned course in graduate exit interviews. The VPS aspect of the course provides students with a realistic introduction to common commercial web development practices. It also reinforces vital concepts for all IT students, whether they plan to pursue web development or another IT career. Finally, the persistent nature of the system provides a showcase for the students and the program.

Bibliography

1. L. Barroso and U. Holzle, "The Case for Energy-Proportional Computing", Computer **40(12)**, 33-37 (2007)
2. Joseph J. Ekstrom, Barry Lunt, "Education at the Seams: Preparing Students to Stitch Systems Together; Curriculum and Issues for 4-Year IT Programs", Proceedings of CITC-4, p. 196-200. (2003)
3. Joseph J. Ekstrom, Barry Lunt, C. Richard Helps, "Education at the Seams: Preliminary Evaluation of Teaching Integration as a Key to Education in Information Technology", Proceedings of the 2004 ASEE Annual Conference & Exposition, session 1450 (2004)
4. M. Tim Jones, "Virtual Linux", IBM developer works, http://www.ibm.com/developerworks/linux/library/l-linuxvirt/, accessed 2/5/09
5. Tom Killalea, "Meet the Virts", ACM QUEUE, January-February 2008
6. R.A. Meyer and L.H. Seawright,"A Virtual Machine timesharing System" IBM Systems Journal, Volume 9, Number 3, Page 199 (1970)
7. Parallels Corp., "Parallels Virtuozzo Containers: Internal and Commercial Hosting", http://www.parallels.com/products/virtuozzo/solutions/hosting/, accessed 2/5/09
8. Pradeep Padala, Xiaoyun Zhu, Zhikui Wang, Sharad Singhal, Kang G. Shin, "Performance Evaluation of Virtualization Technologies for Server Consolidation", HP Labs Technical Report HPL-2007-59R1, http://www.hpl.hp.com/techreports/2007/HPL-2007-59R1.html, accessed 2/5/09, (2007)
9. Mendel Rosenblum, "The Reincarnation of Virtual Machines", ACM QUEUE, July-August 2004
10. M Rosenblum and Tal Garfinkel, "Virtual Machine Monitors: Current Technology and Future Trends", IEEE Computer, May 2005
11. Jack Santos, Burton Group, "A CIO's view of Server Virtualization", http://www.burtongroup.com/Client/Research/Document.aspx?cid=1254 (fee) accessed 2/5/09 (2007)
12. J.E. Smith and Ravi Nair, "The Architecture of Virtual Machines", Computer, May 2005
13. Chris Wolf, Burton Group, "OS Virtualization: Opportunities and Challenges", http://www.burtongroup.com/Client/Research/Document.aspx?cid=1208 (fee) accessed 2/5/09 (2008)
14. Chris Wolf, Burton Group, "Let's Get Virtual: A Look at Today's Server Virtualization Architectures", http://www.burtongroup.com/Client/Research/Document.aspx?cid=1088 (fee) accessed 2/5/09 (2007)