Proceedings of the 14th Colloquium for Information Systems Security Education
Baltimore Marriott Inner Harbor
Baltimore, Maryland June 7 - 9, 2010

Using Virtualization for Cyber Security Education and Experimentation

Todd R. Andel, Kyle E. Stewart, and Jeffrey W. Humphries, *Air Force Institute of Technology*

Abstract – Developing realistic cyber training environments enables hands-on training of cyber security topics in a controlled fashion. Cost, space, time, and reproducibility are major factors that prevent large-scale network replications for individual training purposes. This paper discusses the ways that existing virtualization technologies could be packaged to provide a more accessible, comprehensive, and realistic cyberspace training and education environment to the individual user. The paper maps ways of merging two existing virtualization methods in order to leverage the unique benefits that each type of virtualization technique provides. The first method, called operating system virtualization, provides a highly memory efficient way to run virtual machines, provided that all virtual machines run the same operating system kernel. The second approach, full virtualization, requires a larger memory footprint, but allows arbitrary operating systems to be virtualized. Combining these two techniques will allow for larger, more diverse training environments for modeling and training in the cyber domain.

Index terms – Information Security Education, Cyber Security, Virtualization

I. Introduction

In 2005, the United States Air Force (USAF) changed its mission statement to include "...to fly and fight in Air, Space, and Cyberspace [1]." The ability to develop and train in this new war fighting domain, *i.e.,* Cyberspace, must now be considered. Providing adequate *training ranges* for cyberspace security training, development and testing, or information assurance education presents a significant challenge. Since it would be dangerous to train or test cyberspace tactics and tools on a live operational network, one must develop a way to emulate cyberspace for such purposes. Cyberspace emulation can be considered a modeling and simulation problem, although viewing it in this context is somewhat of a paradox. That is, a computer (or computer network) is used to model a computer (or computer network).

So how do you model cyberspace? The solution can range from a simple computer, a separate off-line training network (or range), or a connection of isolated ranges over live networks using isolation technologies such as a virtual private networks (VPNs). Each approach has its own purpose and provides a solution to modeling a cyber environment.

This paper addresses the different approaches to modeling cyberspace, focusing primarily on utilizing virtualization techniques to provide a small physical footprint, or even stand-alone, training platform to educate cyber security concepts.

II. Modeling Approaches

It is obvious that one would not choose to develop and test cyber security technologies on live operational networks due to the potential impact of unintended side effects. For instance, you would not train network personnel how to build a mail server or how to detect and eradicate a live virus or worm on an operational network. Also, how can new defensive software tools be tested against anticipated threats on a live network. This dilemma leads to having to model the cyber (or network) environment/domain.

There are various approaches available to model the cyberspace domain, such as replicated networks and stand-alone single machine platforms. Each solution has its own advantages and disadvantages.

A. Replicated Networks

Replicated networks are just that, replicated portions of networking infrastructure and end points to effectively test and train cyber related effects. In some scenarios, it is desirable to connect distributed replicated networks to increase scale and add realism. This interconnection can be implemented via isolated network overlays over current operational networks that are already in place, separated by a layer to provide an isolated cyber range. This approach can be accomplished through a simple VPN implementation and has been successfully used in large-scale exercises. Two such examples are the Cyber Defense Exercise (CDX) and the Simulator Training Exercise (SIMTEX), both supported by the United States Department of Defense (DoD).

The SIMTEX [2,3] (recently known as the Joint Cyberspace Operations Range - JCOR) network provides

The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

ISBN 1-933510-99-4 ©2010 CISSE

for large scale experimentation as depicted in Figure 1 [4]. SIMTEX was developed by EADS North American Defense Security and Systems Solutions to support network training requirements for the Air Force through the Air Force Communications Agency.

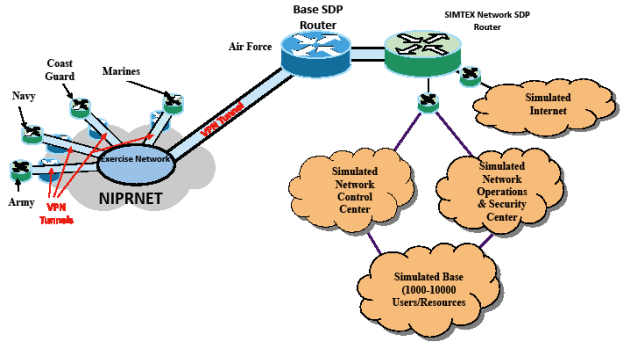

Figure 1. SIMTEX Network

SIMTEX provides a standard environment to emulate large-scale networks over VPN access, without harming the underlying operational network. SIMTEX provides a way to quickly reset this isolated network to a base state after it has been *compromised*, allowing for a repeatable training platform. This solution allows the USAF to train network operator tactics against real-world threats. However, there are a few limitations. First, realistic network traffic must be generated to provide enough background traffic to ensure the attack signal-to-noise ratio does not in itself give it away. There are various options to generate such traffic, which is beyond the scope of this paper. Regardless of the traffic generation option, it is vital that generated network traffic is realistic by using standard stateful session traffic (*e.g.,* emails, web surfing, etc.) and not simply raw bits. Another limiting factor is the extensive infrastructure that must be in place, to include equipment as well as support personnel. Currently there are only 14 permanent nodes within the SIMTEX backbone [3].

Similar to SIMTEX, the National Security Agency (NSA) sponsored CDX provides a large-scale information assurance capstone project for various DoD service academies and graduate schools [5, 6]. For the CDX, the students in these academic institutions build and defend a network against open-source attacks from the NSA. VPN's are also used to attach the remotely distributed networks; however, the NSA Red-Team is the only authorized attacker in this controlled exercise. The intent is to immerse the students in a broad range of activities required to secure and defend a network, prior to heading to operational military units. CDX provides a great educational platform, but also has similar realistic traffic generation, equipment, and personnel requirements as SIMTEX.

B. Stand-Alone Platforms

While events such as SIMTEX and CDX provide great training and educational opportunities, there still exists a requirement for providing a stand-alone platform option to educate and train cyber warriors. For instance, how can one effectively train students or personnel basic network administration activities prior to obtaining the qualifications to participate in large training exercises such as SIMTEX? Textbooks and documentation only go so far, and are not able to provide the added benefit gained by actual hands-on experience. Clearly it is not feasible to offer all personnel their own replicated physical network and it is too risky to offer hands-on cyber training on operational networks.

A solution to this problem lies in providing a stand-alone training platform. The problem with this solution is, "How do you make this a realistic environment?" Platform virtualization is the key enabler to providing such a solution. While virtualization can increase the effective network footprint of the large replicated network approach, it is the vital piece to transform a stand-alone machine into a realistic network training platform. The remainder of this paper focuses on providing such a solution.

III. Towards a Solution

Virtualization is an idea that has been around for a long time, stemming from virtual memory concepts [7]. As desktop computers have become more powerful in recent years, tools for creating virtualized x86-based platforms (*i.e.,* platform virtualization) have matured significantly. There are numerous commercial and open source products that provide near native speed emulation of x86 hardware running on an x86-based host.

A. Virtualization Approaches

Platform virtualization can be generally broken into three categories: *emulation/full virtualization, paravirtualization, and operating system virtualization* [8, 9].

Emulation, or dynamic recompiling, is the process of translating the machine-level instructions of one architecture to another. For instance, emulation allows code written for a PowerPC processor to run on an x86 processor. Since this translation occurs at runtime in software for each instruction, the performance penalty can be significant. When emulating a platform on top of itself (*e.g.,* an x86 platform on an x86 processor), the hardware can execute the majority of instructions directly without translation, resulting in near native performance. This

process is known as hardware assisted emulation or full virtualization.

Paravirtualization runs a thin layer of operating system software called a hypervisor between a *modified* guest operating system and the actual host platform hardware. This process can produce better performance when compared to full virtualization, especially in the case where multiple guests are running simultaneously.

Operating system virtualization uses a shared system kernel to isolate and manage resources in such as way that special user processes, or threads, are made to act like independent machines. Since all virtual machines share the same system kernel, each virtual machine must run the same operating system (*e.g.,* Linux on Linux, Windows on Windows).

B. Solution Requirements

To provide an effective solution one must balance the trade-offs between the benefits and limitations associated with each type of virtualization technology.

Emulation offers the greatest flexibility in terms of what guest platforms can be supported on a given platform. The cost of this flexibility is speed, since each emulated instruction must be translated to the machine code of the underlying host.

Full virtualization maintains the speed of near native performance, but each virtual machine has an equivalent memory, or Random Access Memory (RAM), requirement as its non-virtual counterpart. This requirement severely limits the number of virtual machines that can be simultaneously run. For example, consider a host with 1024 MB of physical memory and a host operating system that has a minimum memory requirement of 256 MB. Only three simultaneous virtual instances of this operating system can be initiated as 768 MB is required by the three guest operating systems and 256 MB is required by the local host. Popular full virtualization software packages include VMware Workstation, Sun VirtualBox, and Kernel-based Virtual Machines (KVM). These software packages are also referred to as hypervisors. The hypervisor manages the interconnections between the guest operating systems and the host operating system or hardware.

Paravirtualization provides better performance than full virtualization, but customized modifications to the guest operating system are required. Examples of this type of virtualization are Xen and User-Mode-Linux.

Operating system (OS) virtualization, such as OpenVZ, is the most scalable technique, supporting over 100 virtual nodes per 1024 MB of physical RAM [9]. However, operating system virtualization does not support a heterogeneous mixture of virtual operating systems.

Further information on various virtualization packages and their capabilities can be found in a paper by Rimondini [9] and in a VMware whitepaper [8].

C. Proposing a Hybrid Approach

The proposed solution in this paper involves implementing a hybrid approach of combining both full and operating system virtualization. An ideal virtual network implementation should provide similar functionality to a comparable real-world network. Specifically, the primary goal is to provide a large virtual network footprint while reducing the required physical footprint to something that the average, modern laptop can handle. The main limiting factor is available RAM in the host platform, referred to as a *RAM budget* for the remainder of this paper.

This solution seeks to realize two primary objectives to support the primary goal of providing a realistic training and testing environment on a small physical footprint. First, the virtual network must support a wide range of operating systems. Second, a large number of concurrent virtual nodes must be supported. An initial goal is to achieve 10-15 nodes per 1 GB of available host RAM, representative of a small network environment with enough resources to provide an adequate virtual training network. The actual number of concurrent nodes will ultimately be determined by the average RAM budget for each virtualized node in the hybrid mixed architecture. Figure 2 provides an overview on this hybrid virtualization model.

This concept requires that the host operating system be capable of supporting both full and operating system virtualization. Linux possesses the ability to support both types. Modern Linux kernels support hardware-assisted full visualization using KVM, utilizing current processor advances such as the Intel VT and AMD-V virtualization extensions. If the associated host is not using compatible hardware, KVM enabled kernels revert to a slower virtual process using the QEMU processor emulation environment that does not require specialized hardware. Merging these technologies to operate concurrently is required to enable a heterogeneous virtual network using a single platform.

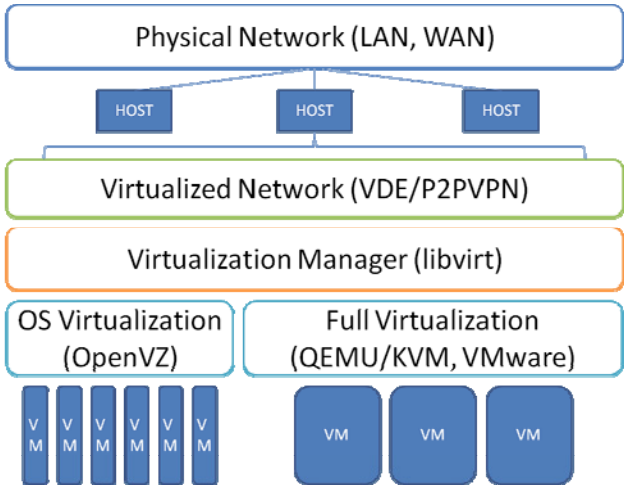

Figure 2. Hybrid Virtualization Model

Hybrid approaches have been previously demonstrated, such as in the Proxmox Virtual Environment (VE) project [10]. Proxmox allows for OpenVZ and KVM combinations utilizing a stripped down, or bare-metal, Debian based operating system. Interaction between the virtual machines is provided via Perl scripts. While PromoxVE is based on open source initiatives, it is currently providing a platform for customized licensed applications such as Promox Mail server to combat spamming techniques.

To support the goals of replicating a small self contained network for training, this paper focuses on hybrid virtualizations using a different approach, relying on the *libvirt* set of virtualization application programming interfaces (APIs) [11] to manage the interface between the operating system and full virtualization models as presented in Figure 2. *Libvirt* also provides networking capability between the virtualized nodes to enable the virtualized network, utilizing connection information written in Extensible Markup Language (XML). The goal of utilizing *libvirt* is to capitalize on the inherent networking APIs to allow interaction between the virtual operating systems on the local hosts. However, the challenge here, as echoed by Proxmox developer Martin Maurer, is *libvirt's* overhead due to its rich API set. This challenge is vital to the RAM budget limitations, but will allow for a more robust virtualized networking capability.

Additional networking capabilities can be provided with the Virtual Distributed Ethernet (VDE) infrastructure [12]. VDE can provide capabilities to network between virtual hosts similar to *libvirt*; however, the desired capability for this concept is to use VDE to provide interconnection between separate physical hosts as depicted in Figure 2. VDE allows the local virtual hosts to be connected to the local physical host, which can then be connected via standard networking technologies to other physical hosts, possibly running their own virtual networks.

Another possibility for virtualizing the network layer is the use of peer to peer-virtual private networking (P2P-VPN) [13] technologies to provide a decentralized approach to building virtual networks. In a traditional centralized virtual private network, machines that are connected across a wide area network (WAN) can communicate as if they were connected to the same local area network (LAN), similar to the setups used in SIMTEX and CDX. The edges of the WAN that wish to connect to the VPN LAN connect to a central server and the clients establish a secure communications channel with the central server. When the client wishes to send a packet to another client on the same VPN LAN, it sends the packet to the central server which relays it onto its destination. In contrast, a peer to peer approach allows clients to create direct secure connections with each other. This approach has the potential to remove the central server as a potential bottleneck for network traffic. It also reduces or possibly eliminates the need for a centralized management and authentication system.

The use of a P2P-VPN allows for a virtual network topology to be defined in software regardless of the underlying physical topology. Therefore, if operating a P2P-VPN across a traditional physical network or utilizing for virtual network connectivity in the same box, the operating systems are unaware and do not have to be modified. Some P2P-VPN software packages such as N2N [14] allow for VPNs at Layer 2. A Layer 2 VPN transports and secures entire network frames. This is in contrast to a Layer 3 VPN such as IPOP [15] that works at the IP transport layer of the TCP/IP stack. The benefit of using a Layer 2 solution is that most hypervisors allow some type of bridging mechanism from the virtual Ethernet adapter inside the virtual machine to an adapter on the host. This additional level of network layer abstraction allows a software defined network topology of virtual machines to be connected together regardless of hypervisor or physical location of the host. This feature provides a great deal of flexibility when building training ranges in terms of scale of the range as well as the physical location of the end nodes (*i.e.,* the students).

While the initial goal of this concept is to provide a stand-alone platform to train and educate cyber technologies, it may also provide long term additional capabilities by allowing separated stand-alone platform virtual networks to connect. This additional capability would allow two students to build independent networks and then connect together to participate in localized red/blue exercises. Ultimately, incorporating the ability to provide independent stand-alone virtual network platforms with the ability to connect independent virtual networks using VDE or P2P-VPNs can be used to significantly increase

the effective network footprint used in the large-scale ranges previously discussed.

D. Scenario Development

Since this concept aims at transforming the large-scale cyber-range exercises towards a stand-alone platform, an efficient distribution method must be provided for the virtual networks. To be an effective cyber training platform, various virtual network images need to be constructed to provide for various network infrastructures. These images may include the entire virtual network, consisting of virtual hosts running applicable software (*e.g.,* mail or web servers), routers and switches, and interconnections. It is vital that any constructed images remain under the RAM budgets of the intended target platform and can be provided via a central server if physical network connections exist, or on external drives. Another advantage to the imaging approach is that if a virtual network gets corrupted it can simply be reloaded with relative ease. This approach allows the evaluation and training of potentially dangerous exploits, malware, and cyber attacks.

In addition to providing generalized virtual network images, a need for specific scenarios may be required in some instances. These scenarios should be customizable to a specific task at hand, but should be capable of reusing generalized virtual network images as much as possible. For example, consider a virtual network image is developed to replicate the information technology infrastructure of a small business. A network for this type of business might contain 15-20 nodes including workstations, servers, routers, firewalls, and other networking equipment. Multiple scenarios can be developed using this virtual network image that target a variety of cyber education tasks. For example, a scenario can put the student in the attacker's mind and walk them through how to penetrate the potentially vulnerable network. After this type of lesson, the student is now prepared to learn how to perform forensics on the virtual network that was just compromised in order to learn how to identify network attacks. Finally, attacks can be scripted to run automatically as the student is tasked with actively defending the network. In this way, students progress through customized scenarios using the same virtual network image in order to gain important hands-on experience.

IV. Initial performance results

Before a realistic training platform can be implemented using a hybrid virtualization approach, it is vital to determine the feasibility and performance of such a system. Performance experiments were accomplished to validate the potential of the approach and provide motivation for further research and investigation into how large of a network can be realized within a given RAM budget. These experiments test the scalability of the different virtualization techniques.

A. Experimental Setup

All experiments were conducted on a Dell PowerEdge 860 server with the following specifications:

- Intel Xeon 3060 Conroe 2.4GHz 4MB L2 with 1066 MHz bus and Intel VT-x virtualization
- 4 GB of RAM
- 2 x 80 GB hard drives
- 2 x Network Interface Cards

The primary hard drive was divided into several logical volumes and each hypervisor was installed into its own operating system per logical volume. The benchmarking tool was the *build-apache* module of the Phoronix Test Suite 2.4.1 [16]. This module measures the length of time required to compile the Apache web server. The host in all cases is 64 bit CentOS 5.4 with no updates. The guests were made to be as similar as possible. In each case, 64 bit CentOS 5.4 was used with the minimal amount of installation necessary to run the benchmarks and communicate with the host. The default configurations were used for both the virtual machines as well as the hypervisor.

The experiments consist of two factors and two metrics. The two factors are the hypervisors (*i.e.,* the virtualization approach) and the number of concurrent virtual machines. The two metrics are the time taken to perform the benchmark on the host and the overall amount of time (*i.e.,* wall time, to include benchmark setup items) taken to complete all the benchmarks. In all cases, a separate instance of the benchmark is run inside the host as well as inside each virtual machine. For example, when running two virtual machines there are a total of three instances of the benchmark executing concurrently.

B. Data Analysis

The first aspect that must be considered is what effect occurs to the host system. Figure 3 shows the impact of increasing the number of virtual hosts has on the host performance using the chosen benchmark. The hybrid approach consists of an equal number of full (VirtualBox) and OS virtualization (OpenVZ) instantiations. While it is not always possible to install two hypervisors at the same time, VirtualBox and OpenVZ do not have any direct conflicts and can be made to operate at the same time.

The data indicates that paravirtualization and OS virtualization rely heavily on the host OS, thus causing a higher impact on the host performance itself. The hybrid approach impacts the host OS closer to that of the full virtualization which has minimal dependencies and impact on the host OS.

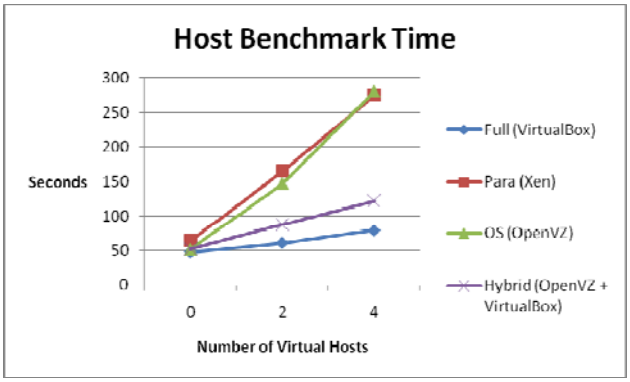

Figure 3. Impact to Host

Full virtualization did not provide a significant impact to the host and can run different operating systems at the same time. However, the total impact on the host and virtual machines, as seen in Figure 4, limits full virtualization uses towards achieving the goal of replicating a small network infrastructure on the same platform.

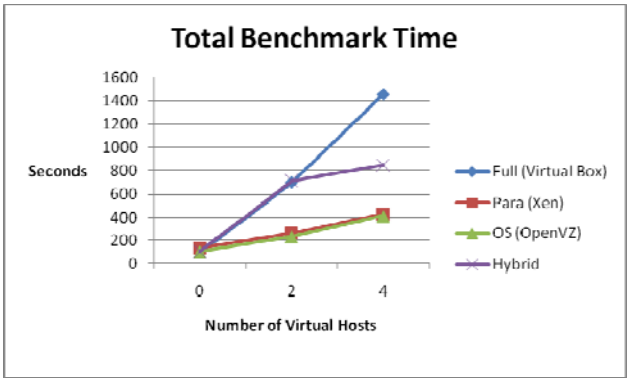

Figure 4. Total Impact

The data suggests that the reliance on the host OS reduces the impact on the virtual machine performance for both paravirtualization and OS virtualization.

When looking at the overall impact on the host and virtual machines, the hybrid approach bridges the gap by providing an achievable mix of impact on the host operating system while at the same mitigating the impact on the virtual hosts. These two aspects are vital to provide the correct approach to achieve a realizable system capable of concurrently emulating multiple heterogeneous operating systems. Future work will isolate this hybrid implementation and focus on RAM budgets and training scenarios.

V. Conclusions

As cyber security technologies are developed, the new challenge becomes how to test, train, and educate personnel or students in a realistic network environment. Due to their risky nature and potential impact to operational networks, cyber security tools and tactics cannot be performed on live operational networks. Large-scale cyber training exercises using replicated networks provide a vital capability, but the coordination, manpower, and equipment costs are not trivial. A stand-alone training alternative would greatly enhance cyber security training and allow personnel the ability to train at their own pace, without interfering with network operations or the training of other personnel.

This paper introduces concepts the authors are currently working towards developing a stand-alone cyber training platform utilizing virtualization techniques. The goal is to create a realistic virtual network footprint onto a small physical footprint, by developing a hybrid approach to combine both full and operating system virtualization onto a single workstation/laptop. The limiting factor is finding the correct virtualization mix to provide a realistic network, but within the RAM budget of the target host. Performance is also a concern that needs to be considered. A usable virtualized network training system must appear to operate as a normal network.

Continual training with cyber security will always exist. Hardware and software continuously changes and the threats faced are dynamic in nature. Providing a platform to keep personnel trained to defeat such threats is vital, whether using a large-scale replicated network or a stand-alone option.

VI. References

[1] Mitch Gettle. Air force releases new mission statement. http://www.af.mil/news/story.asp?storyID=123013440December 2005. Date Accessed: April 20, 2009.

[2] EADS NA Defense Security and Systems Solutions. Simtex security training and development. http://www.eads-nasecurity.com/coast simtex solution.html. Date Accessed: April 22, 2009.

[3] Aaron McBride. Air Force cyber warfare training. *Defense Standardization Program Journal*, pages 9–13, April/June 2007.

[4] Jeffrey Tibbitts. Risk management – The Department of Defense information technology security certification and accreditation process (DITSCAP) model. http://louisville.edu/infosec/Fall Date Accessed: April 22, 2009.

[5] W.J. Schepens and J.R. James. Architecture of a cyber defense competition. In I*EEE International Conference on Systems, Man and Cybernetics (IEEE SMC 03)*, volume 5, pages 4300–4305, Oct. 2003.

[6] B.E. Mullins, T.H. Lacey, R.F. Mills, J.M. Trechter, and S.D. Bass. How the cyber defense exercise shaped an information-assurance curriculum. *IEEE Security & Privacy*, 5(5):40–49, Sept.-Oct. 2007.

[7] W.C. McGee. R68-49 virtual memory processes and sharing in multics. *IEEE Transactions on Computers*, C-17(11):1099, 1968.

[8] VMware. Understanding full virtualization, paravirtualization, and hardware assist introduction. http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf, 2009. Date Accessed: May8, 2009.

[9] Massimo Rimondini. Emulation of computer networks with netkit. Technical Report RT-DIA-113-2007, Universit‘a di Roma Tre, January 2007.

[10] Proxmox. Main Page - Proxmox VE. http://pve.proxmox.com/wiki/Main_Page. Date Accessed: May 4, 2009.

[11] Libvirt. The virtualization api. http://libvirt.org/index.html. Date Accessed: May 4, 2009.

[12] Renzo Davoli. VDE: Virtual distributed ethernet. In *First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*. Tridentcom 2005., pages 213–220, 23-25 Feb 2005.

[13] David Isaac Wolinsky, Linton Abraham, Kyungyong Lee, Yonggang Liu, Jiangyan Xu, P. Oscar Boykin, Renato J. O. Figueiredo: On the Design and Implementation of Structured P2P VPNs, *CoRR-abs/1001.2575*, 2010.

[14] n2n: a Layer Two Peer-to-Peer VPN. http://www.ntop.org/n2n/. Date Accessed: Mar 6, 2010.

[15] IPOP (IP over P2P). http://www.grid-appliance.org/wiki/index.php/IPOP Date Accessed: Mar 6, 2010.

[16] Phoronix Test Suite - Linux Testing & Benchmarking Platform. http://www.phoronix-test-suite.com. Date Accessed: April 16, 2010