The Efficacy of Live Virtual Machine Migrations Over the Internet

Eric Harney, Sebastien Goasguen, Jim Martin, Mike Murphy, Mike Westall
School of Computing, Clemson University
eharney, sebgoa, jim.martin, mamurph, westall@cs.clemson.edu

ABSTRACT

This paper describes a technique to enable live migration of virtual machines over the Internet. The method assumes the network supports Mobile IPv6 and that participating host devices support Xen 3.1. Unlike live migration schemes proposed by other researchers, virtual networks are not required. We describe our work in progress on developing a system that utilizes Mobile IPv6 to enable constant network connectivity through the migration. We identify the sources of delay associated with the live migration and conclude that as long as migrations occur relatively infrequently, live migration over the Internet is practical.

1. INTRODUCTION

Virtual Machine (VM) technology was originally developed by IBM in the 1960's to support their System 360 mainframe class of computers [1]. A Virtual Machine Monitor (VMM) is system software that provides users the appearance of direct access to a dedicated machine. With advances in technology and economics, the use of VM technology has proliferated. The dominant VM product vendor, VMware, provides a complete virtualization of Intel and AMD based PCs [2]. Xen, an open source virtualization project, provides a '*para-virtualization'* that, up until recently, has required modifications to the guest operating system to support a VM abstraction. [3] It is now possible for Xen to support unmodified guest machines by utilizing hardware support (Intel has contributed modifications to Xen to support their VT-x architecture extensions and AMD has contributed support for their AMD-V extensions).

As VM technology becomes ingrained in the IT fabric, managing VMs is crucial. VMware and Xen now have centralized capabilities to monitor and manage VMs that are deployed. The next step is 'live migration', where a VM is moved from one physical machine to a different machine. The challenge is to provide live migrations without impacting applications that are running on the VM. Live VM migration is now supported by Xen and VMware, but only in scenarios where migration occurs on the same LAN. The research community is developing schemes to support Wide-area Overlays of Virtual Workstations (WOWs) that can provide virtual computing environments over the Internet [4]. A cumbersome issue is dealing with the network deployment context required to preserve end-to-end connectivity as VMs are migrated to new physical machines that are likely located on different networks [5, 6, 20]. To the best of our knowledge, all proposed ideas rely on 'virtual networks' (i.e., overlay networks) to facilitate the interconnection of VMs. We take a radically different approach. Using standard IPv6 protocols, we use mobile IP to provide seamless network connectivity to support the migration of VMs over the Internet.

Our research examines the efficacy of live migration over the Internet. We focus on what we consider to be the two major obstacles: moving the network state of the VM from one location to another without loss of service, and maintaining access to the VM's virtual disk. While Mobile IPv6 was designed to support mobile wireless IP devices, it also provides a convenient method for establishing the network context for a VM that is migrated to a new physical machine. Assuming Mobile IPv6 is widely deployed in the future Internet, our approach to live migration eliminates the need for non-standard and complex virtual networking schemes to interconnect the VM platforms.

This paper is organized as follows. First we summarize motivations for live migration over a WAN. Next we identify the main obstacles followed by a discussion of related work. We then describe our proposed solution based on Mobile IPv6. We describe our evaluation methods and summarize results. Finally we present conclusions and identify future directions.

2. MOTIVATION FOR WAN MIGRATION

Live virtual machine migration over a WAN is a relatively new concept. While the initial benefits are likely to be derived from increased reliability and availability of VMs, there are other interesting applications. Transporting compute environments as appliances provides new opportunities for grid computing systems. Since an entire environment based on a virtual machine can be distributed easily across a heterogeneous collection of hosts, grids can be assembled out of any machines capable of hosting the VMs. If we enable WAN migration with persistent network state for these VMs, it becomes possible to dynamically reconfigure a grid by migrating virtual machines without actually interrupting their service or causing additional overhead. This allows the grid to be reconfigured in response to specific demands or load balancing needs. A case for using virtual machines in grid computing has been made in [7] and a web service interface to Xen has been developed in [8].

The Internet Suspend/Resume project has developed a system which allows a virtual machine to be migrated over the Internet to another physical system [9]. This provides the ability for a user to go to any computer which is connected to the Internet and migrate their standard VM to that machine – providing the same environment and data for their use, in all places. Enhancing this system to include network state persistence across migrations would facilitate networking for the VM, since it could maintain the same address at all locations as well.

Figure 1 illustrates the concept we are developing. The resource manager accepts requests for system resources from users and initiates the creation or selection of an “appliance”[1]. The resource manager assigns an appliance to one or more VM platform nodes, and over time, moves the appliances to different VM nodes based on performance, policy, or availability. The scope of the research reported on in this paper is limited to evaluating the practicality of migrating appliances in a live manner over the commodity Internet.

3. THE PROBLEM

There are two pieces of state associated with a VM. Each VM has a ‘virtual hard drive’ that is maintained by the VMM as a (large) flat file. The VMM attaches this file to the VM as a block device, and the VM’s OS uses it as if it is a physical hard drive. Secondly, a running VM has a network state, which includes its IP addresses and all currently open connections.

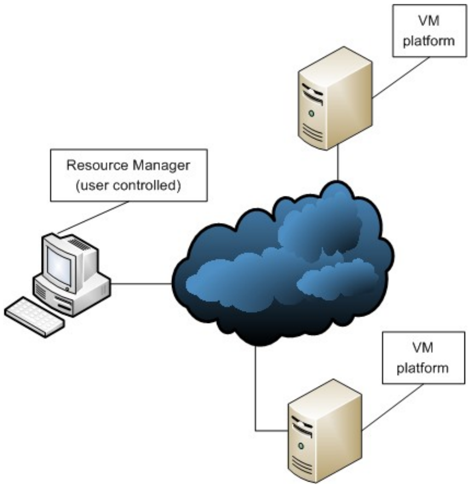

Figure 1: System description

When migrating a virtual machine to another host on the same LAN, one can simply mount a networked or distributed file system on the destination host. When a virtual machine migrates to this host, it restores its state; the VM is able to find its virtual hard disk by simply looking in the same path on the destination VM host as it did on the original VM host.

However, when migrating a virtual machine across a WAN, the situation changes dramatically. In most cases, two servers on separate networks may not have direct access to the same NFS server or cluster file system. Even if they did have access to the same NFS server, the latencies introduced by the virtual disk transfer over a WAN such as the Internet are likely to be significant.

A further issue with which to deal when migrating a virtual machine is that of network connectivity and state. A VM typically has one or more IP addresses assigned to it, and any number of network connections open at a given time. In order for a truly “live” migration to happen, the address(es) and connection state must be preserved when the VM is migrated between hosts. Merely migrating the virtual machine and maintaining the same addresses is not sufficient, as service will be interrupted for any clients connected to the virtual machine.

A system to support live migration over a WAN should have the following qualities:

Continuous service: Any services provided by the virtual machine should sustain minimal downtime and interruption when the virtual machine is migrated.

Permanent address for the VM: The network address of the virtual machine should not change after migration occurs.

Interoperability with IP protocols: A virtual machine should be able to be migrated over a WAN connection without interfering with application-level communications. TCP and UDP should behave normally, with minimal side effects introduced by the migration or virtualization system, so that clients accessing the virtual machine are not affected. Additionally, the migration should be able to occur between any two hosts on the Internet.

Moving the address and network state of a virtual machine is a relatively simple proposition if the source and destination VM hosts are operating on the same LAN subnet. The Xen platform has had this capability for some time. [10] In this case, the virtual machine can keep its IP and still have proper routing when migrated to another host machine. It must, however, update its link-layer location, which is accomplished by sending a gratuitous ARP update with its IP and the MAC address of the new Xen host to the network. This operation is quick enough to keep the downtime in the tens of milliseconds.

This method of migration on a LAN meets the above requirements, because it provides a very short interruption of service, equivalent to the time it takes for ARP to update the switch with the VM's new location. The address of the VM is never changed since the VM stays on the same network, and keeps its IP address. Additionally, no extra protocols are required because the network migration occurs at the link layer with ARP.

[1] We use appliance in the same manner as a ‘virtual appliance’ which was defined in [21] to be an application combined with an environment needed by the application to execute in a virtual environment.

4. PREVIOUS WORK

Systems have been designed which optimize the way the disk image for a VM is transferred when migrating across a WAN. [11] This enables significantly reduced downtime compared to a system which just copies the disk image. The system described in [11] performs a bulk transfer of the virtual machine image before the actual migration takes place, and then transfers small deltas of the changing parts of the disk. This allows for a substantial reduction in the amount of downtime caused by the time it takes to transfer the VM image during migration. This work provides a good baseline for what can be achieved with migrating the disk image.

The same paper provides a way to migrate the network state of a virtual machine when it moves to a different network. A combination of dynamic DNS and IP tunneling is used to maintain connections to the VM. At migration time, an IP tunnel is created using iproute2 which forwards packets from the old VM host to the new host for the VM. Dynamic DNS is used to update the IP address listed for services on the virtual machine, so that new connections can bypass the tunnel and locate the VM at its new destination.

P2P networks have been used to create virtual LANs [4], presenting another way to handle the problem of maintaining network connectivity to a migrating virtual machine. The system proposed in [4] uses a P2P network to create a virtual LAN over different networks. This provides advantages over the approach of using DNS and tunneling, because it allows a virtual machine to keep the same IP. Since the IP does not change when a machine is migrated, existing TCP connections can stay open. [5] also provides a different approach, where network connectivity is handled using DHCP to build networks in a decentralized system.

Standard IP tunneling has also been used to maintain network connectivity after migration. [12] The system designed with this method also demonstrates that low downtime can be achieved when migrating over a WAN. This provides a baseline for performance our system should be able to achieve.

[13] describes another method of creating virtual networks using an overlay to create a virtual LAN. In this project, nodes are combined into virtual LANs with virtual hosts and virtual routers. This creates layers of isolation above the Internet itself on which the VMs communicate.

The advantage of using Mobile IPv6 to support live VM migration is twofold: since the virtual machine's IP address does not change, DNS updates are not needed to locate services on the VM. It remains accessible at the same logical address at all times. Secondly, Mobile IPv6 provides the ability to use route optimization when contacting a machine that is on a foreign network, rather than tunneling. This allows connections to happen directly to the machine without relying on a higher-level protocol such as DNS to locate it.

5. PROPOSED SOLUTION

The objectives of our research are similar to those of [11] and [4]. However, rather than using an overlay or dynamic DNS, we propose a solution based on the standard Mobile IPv6. The resulting network configuration provides capabilities similar to [12], but offers two advantages over their IP tunneling system: tunnel setup is handled automatically by the VM when needed, and hosts supporting Mobile IPv6 can use route optimization mode to bypass the tunnel when connecting to the VM.

The Mobile IPv6 standard [14] enables a machine to maintain the same IPv6 address when moving between networks. This is done by creating a dynamic binding between a machine moving between networks and its home network. As the machine moves between networks, packets sent to its original address are either tunneled or rerouted to its new location.

Mobile IP was originally developed with the mindset of using it for portable computing devices. A typical use case for this is a user who travels between home and work with a laptop.

In Mobile IPv6, there are three types of nodes. A mobile node is the machine that travels between networks. This mobile node is configured to have a “home” network which is its usual location, as well as a home address on this network. A home agent node is run on this network. As the mobile node moves between networks, it maintains contact with its configured home agent. A binding is created that links the home agent with the mobile node's home address to its current (care-of) address.

A node communicating with the mobile node is known as the correspondent node. The interaction between the correspondent node and the mobile node while it is away from “home” can take place in two ways. If the correspondent node supports Mobile IPv6, route optimization can be performed, allowing the correspondent node to talk directly to the mobile node and bypass the mobile node's home network. If the correspondent node does not support Mobile IPv6, packets from the correspondent node are captured by the home agent and tunneled using IPv6 encapsulation to the mobile node.

Since our design objective is interoperability with the future commodity Internet, we make several assumptions. First, the capacity of the Internet will steadily increase allowing hosts direct access to multiple gigabits per second of bandwidth. Second, the IPv6 will be universally deployed. ARIN now advises that projects seeking new blocks of Internet addresses should now choose IPv6 [15]. Further, we assume that Mobile IPv6 will be widely deployed. Lastly we assume that participating hosts support a standard VM platform that is capable of live migrations.

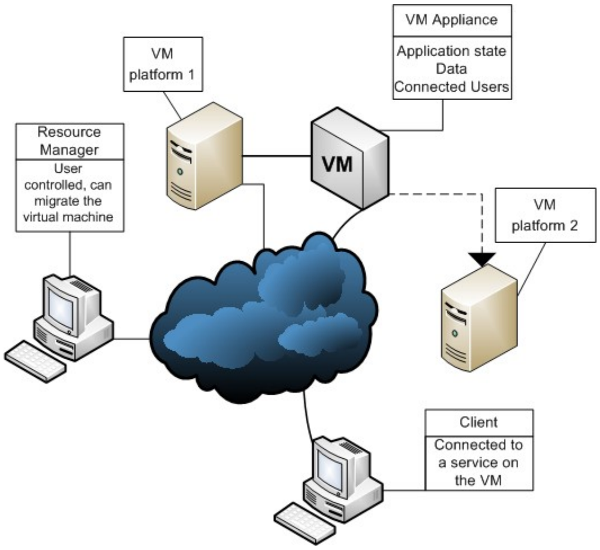

Figure 2: Proposed solution

Figure 2 illustrates our proposed solution. As an example, a client application (that could be running on a VM or on another host connected to the Internet) interacts with a server application running in VM Appliance 1 which physically runs on VM Platform 1. The resource manager decides to migrate the VM Appliance to VM Platform 2. This happens without the client losing connectivity to the service provided by the Appliance.

6. METHODOLOGY

6.1 Lab Environment

The environment we have constructed for testing Mobile IPv6 with virtual machines consists of two Linux Xen 3.1 hosts, connected by a Linux-based router. All network connections are gigabit Ethernet.

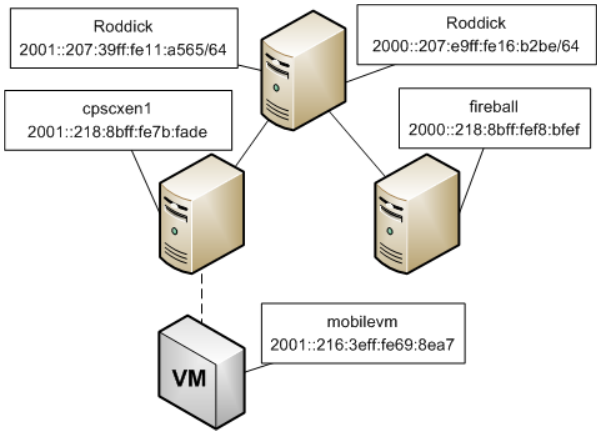

Figure 3: Lab environment

Roddick acts as a router between the two networks, and uses Linux's netem [16] to induce variable delays between the two networks. This allows us to simulate the interaction of virtual machine hosts over a long-distance connection.

Each Xen host, cpscxen1 and fireball, act as a router for the VMs on their network. Each broadcasts ICMPv6 router advertisements using radvd [17,18], an IPv6 router advertisement daemon. This allows a VM to use IPv6 stateless autoconfiguration, and in turn allows mip6d [19] (the Mobile IPv6 daemon) to detect which network it is on.

Cpscxen1 runs mip6d 0.4-rc1 configured as the home agent for Mobile IPv6. Radvd is configured to send router advertisements every 0.07 seconds, the recommended setting when using Mobile IPv6. Mobilevm runs mip6d 0.4-rc1 and is configured as the mobile node.

Cpscxen1 and mobilevm run custom-built kernels, which are based on Fedora Core 7's 2.6.21 Xen Linux kernel. They were built with the USAGI mipv6 umip-0.3 patch [19] applied, which supplies the kernel portion of the Mobile IPv6 stack.

A virtual machine (mobilevm) is migrated between cpscxen1's network and fireball's network using Xen’s built-in live migration feature with the command “xm migrate --live mobilevm fireball”.

6.2 Performance Metrics

Experiments are designed to simulate an application running on a virtual machine which is migrated over a WAN. The basic structure of the tests is to run an application on a VM, migrate the VM, and determine the amount of downtime the application experiences as a result of the migration by analyzing packets captured between the virtual machine and another host.

We developed a simple program that sends a stream of UDP packets from a client to a server in a manner that emulates a G.711 VoIP stream. The tool sends a 210 byte packet every 20ms for a configured amount of time. Once a target number of packets have been sent, the program computes latency, jitter, and loss rate summary statistics. This method provides a convenient mechanism to assess the downtime caused by live migration. The tool was modified to operate in an IPv6 network.

We are interested in the downtime that an application will experience when its VM is migrated between two networks. To determine the exact downtime, we run a server with the UDP tool on mobilevm and a client on cpscxen1 (simulating another host on the WAN). Running tcpdump on mobilevm then allows us to observe the gap in the stream of packets from cpscxen1 to determine the amount of time that the VM is not accessible at its home address. We define this as the “downtime” that a client connecting to a service on the VM would experience.

7. PERFORMANCE ANALYSIS

We conducted two experiments, which demonstrate two separate cases of using Mobile IPv6 for migrating VMs over a WAN. The first is meant to measure the amount of downtime an application will experience in a scenario where both Xen hosts have connectivity to the VM image. The downtime here consists of the time it takes for the Mobile IPv6 stack to detect that it has moved to a new network, establish a new binding with the home agent, and begin routing packets to the virtual machine, in addition to the small downtime always present in the final stages of Xen live migration. Since the VM is running mip6d, it can be migrated as usual with “xm migrate”, and bind with the Mobile IPv6 home agent (cpscxen1) to activate its original address again. This operation uses Mobile IPv6 as an additional feature configured on top of Xen’s migration capabilities. The live migration of CPU and memory state is handled by Xen the same as they are on a LAN.

The second experiment measures a migration in which the full virtual disk is moved between two Xen hosts. This type of migration allows a VM to migrate between two Xen hosts with no shared storage. The downtime experienced is far greater than the previous case due to the large amount of data transferred, but this setup has the advantage of supporting any Xen host on the Internet that supports Mobile IPv6 and does not require overlays or additional infrastructure. It may be possible to optimize this technique by establishing a standard “base” VM image for a given OS, and only transferring deltas in relation to this base to move an OS.

Using these two different techniques helps provide insight into the performance of migrating VMs over a WAN in different scenarios. Experiment 1 measures best-case performance in a situation where the two networks have a shared storage medium. However, in some cases, it is ideal to be able to move the entire VM to another network without this requirement. Experiment 2 looks at this case.

7.1 Experiment 1 – Migration with Shared Storage

The goal of this experiment is to measure the downtime experienced in a stream of traffic when a virtual machine migrates to a foreign network. The magnitude of this downtime helps determine what types of services the Mobile IPv6-based migration system can be successfully used with. The test provides insight into the small transition period when the virtual machine is migrated and switches from one network to another.

The application-level downtime for the VM was calculated by capturing packet dumps on mobilevm while migration takes place. For a short period of time, the UDP packets will be dropped. The sequence numbers embedded in the UDP packets by the UDP tool let us calculate the number of packets lost, and therefore the amount of network downtime the virtual machine experienced.

The network delay is the total delay between cpscxen1 and fireball. To generate the network delay, half of the desired delay is applied to the interface on Roddick on cpscxen1's network, and the other half is applied to Roddick's interface on fireball's network. Multiple network delays are tested, which replicate different possible scenarios that would be experienced on the Internet.

The downtime experienced by the application is caused by the series of events which happen as Mobile IPv6 switches from one network to another. The VM must first receive a router advertisement to learn that it has changed to a new network. A binding update is sent to the home agent’s (cpscxen1’s) mip6d. It performs duplicate address detection to ensure that the requested home address is not in use, and replies with a binding acknowledgement. This allows mobilevm’s mip6d to establish a binding and setup tunneling between itself and cpscxen1.

The results from experiment 1 are illustrated in Figure 4. These measurements were gathered from an average of three runs per data point. There is a minimum downtime of about 2.5 seconds, which scales up linearly as the delay between the networks is increased.

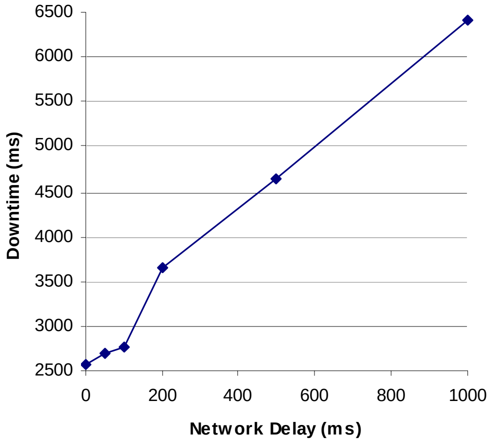

Figure 4: Experiment 1 results

While this size of delay not ideal for an application such as VoIP, it is something that can be handled and recovered from by TCP, especially for less interactive applications.

7.2 Experiment 2 – Migration without Shared Storage

For this experiment, a virtual machine and its image are moved between cpscxen1 and fireball. This is done by saving the VM's memory state to a file, copying the memory state and disk image over to fireball, and restoring the VM on fireball. The disk image for the virtual machine is 4GB, and the memory size is 256MB. Additional network delay is not induced by Roddick for this experiment, as the downtime from copying the image is large enough that the additional delays caused during Mobile IPv6 binding are not as significant.

A virtual machine is moved from cpscxen1 to fireball using the following commands:

```
xm save mobilevm /tmp/mobilevm_state
scp /var/lib/xen/images/mobilevm/vm1
        fireball:/var/lib/xen/images/mobilevm/
scp /tmp/mobilevm_state fireball:/tmp
ssh fireball xm restore /tmp/mobilevm_state
```

To test the amount of downtime, ping6 is used to ping the mobilevm. It goes offline with the “xm save” command, which pauses the VM and dumps its state to a file. After the transfer, the virtual machine is restored (“xm restore”) and comes back online in the same state it was on cpscxen1.

For this configuration, the virtual machine can be migrated from one host to another and restarted in 181 seconds. This may be broken down into five components: the time to save the VM's state; time to copy the VM disk image to the destination machine; the time to copy the VM's memory state to the destination machine; the time to restore the VM on the destination machine; and the time for mip6d to detect it is on the new network, and bind again to the home agent. Results from this experiment are shows in Table 1, and the relative contribution of each component of delay is illustrated in Figure 5. These measurements represent an average of three runs.

When the VM is restarted on the destination machine, its Mobile IPv6 daemon comes up in the same state it was in on the source host. At this point, it still has the same address and network information that it had on the previous host. Upon receiving an IPv6 router advertisement, the VM will use IPv6 autoconfiguration to add a new IPv6 address known as the care-of address to its primary network interface. The mip6d daemon will recognize this, and detect that it is now on a foreign network. When this happens, the VM will send a binding update to its configured home agent (cpscxen1). The home agent sends a binding acknowledgment back to the VM and configures the necessary Mobile IPv6 tunnel.

Operation	Time (seconds)
Save	6
Image copy	126
Memory copy	13
Restore	1
IP Mobility	8

Table 1: Experiment 2 results

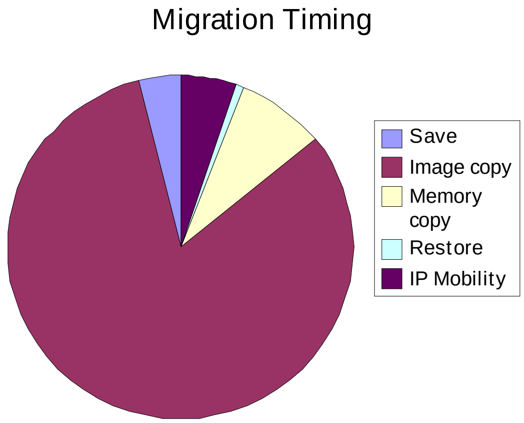

Figure 5: Experiment 2 results

This provides a baseline performance for migrating a virtual machine with its image over a WAN connection. More importantly, the experiment demonstrates that it is possible to maintain IP-level connectivity when migrating a virtual machine over two separate networks.

Since both Xen hosts support Mobile IPv6, when the machine is brought up on the destination host, it is able to autoconfigure its IPv6 address, and establish a new binding with the home agent.

8. CONCLUSIONS

We have presented a method for migrating virtual machines over the Internet using Mobile IPv6. In our lab, we have demonstrated that a VM can be migrated without causing loss of end-to-end connectivity. Our experiment using shared storage suggests that a migration, in the best case, takes on the order of seconds. Our experiment without shared storage suggests that copying the image is one of the major components in downtime in a system which migrates the full virtual disk state. Integrated support in Xen to minimize the VMs downtime will significantly reduce the impact of the VM image copy.

Using Mobile IPv6 provides a standards-based approach for preserving the network context once a VM is migrated. A live migration over the Internet will likely have downtime orders of magnitude longer than live migration involving nodes in a single LAN. However, as long as migrations are infrequent, and as long as end-to-end connectivity can be maintained, live migration over the Internet is possible and offers potential benefits that should offset performance issues.

In our work, we have not considered important practical concerns such as NAT, organizational boundaries, and security. In future work we plan on addressing these issues. In addition, we will explore how our system can support a dynamic environment where VM platforms join and exit the system frequently.

9. References

[1] R. Goldberg, “Survey of Virtual Machine Research”, IEEE Computer, pp. 34-45, June 1974.

[2] VMware. Virtualization: Architectural Considerations and Other Evaluation Criteria, available online at: http://www.vmware.com/pdf/virtualization_considerations.pdf

[3] Boris Dragovic, Keir Fraser, Steve Hand, Tim Harris, Alex Ho, Ian Pratt, Andrew Warfield, Paul Barham, and Rolf Neugebauer. Xen and the Art of Virtualization. In Proceedings of the ACM Symposium on Operating Systems Principles, October 2003.

[4] Ganguly, A., Agrawal, A., Boykin, P.O., Figueiredo, R., "WOW: Self-Organizing Wide Area Overlay Networks of Virtual Workstations," Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing, 2006, pp. 30-42, June 19-23 2006

[5] Ganguly, Arijit, David Wolinsky, P. O. Boykin, Renato Figueiredo 'Decentralized Dynamic Host Configuration in Wide-Area Overlay Networks of Virtual Workstations'. In Workshop on Large-Scale and Volatile Desktop Grids (PCGrid), 03/2007

[6] A. Ganguly, A. Agrawal, P.O. Boykin, R.J. Figueiredo. 'IP over P2P: Enabling Self-configuring Virtual IP Networks for Grid Computing', In Proceedings of 20th IEEE International Parallel & Distributed Processing Symposium (IPDPS)

[7] R. Figueiredo, P. A. Dinda, J. A. B. Fortes, "A Case for Grid Computing on Virtual Machines", Proc. International Conference on Distributed Computing Systems (ICDCS), May 2003.

[8] Keahey, K., T. Freeman, J. Lauret, D. Olson. “Virtual Workspaces for Scientific Applications” SciDAC 2007 Conference, Boston, MA. June 2007

[9] Kozuch, M., Satyanarayanan, M., Internet Suspend/Resume, Proceedings of the 4th IEEE Workshop on Mobile Computing Systems and Applications, Callicoon, NY, June 2002

[10] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live Migration of Virtual Machines. In Proceedings of the 2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI), Boston, MA, May 2005.

[11] Bradford, R., Kotsovinos, E., Feldmann, A., and Schiöberg, H. 2007. Live wide-area migration of virtual machines including local persistent state. In Proceedings of the 3rd international Conference on Virtual Execution Environments (San Diego, California, USA, June 13 - 15, 2007). VEE '07. ACM Press, New York, NY, 169-179.

[12] Travostino, F., Daspit, P., Gommans, L., Jog, C., de Laat, C., Mambretti, J., Monga, I., van Oudenaarde, B., Raghunath, S., and Wang, P. Y. 2006. Seamless live migration of virtual machines over the MAN/WAN. Future Gener. Comput. Syst. 22, 8 (Oct. 2006), 901-907.

[13] Jiang, X., AND Xu, D. Violin: Virtual internetworking on overlay infrastructure. Tech. Rep. CSD TR 03-027, Department of Computer Sciences, Purdue University, July 2003.

[14] D. Johnson, C. Perkins, J. Arkko, Mobility Support in IPv6, IETF RFC 3775, June 2004. Available online at: http://www.ietf.org/rfc/rfc3775.txt?number=3775.

[15] Plzak, Raymond A. ARIN Board Advises Internet Community on Migration to IPv6, May 21, 2007. Available at http://www.arin.net/announcements/20070521.html.

[16] Hemminger, Stephen. Network Emulation with NetEm, April 2005. LinuxConf AU 2005.

[17] Available online at http://www.litech.org/radvd/.

[18] T. Narten, E. Nordmark, W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", IETF RFC 2461, Dec. 2998. Available at http://www.ietf.org/rfc/rfc2461.txt.

[19] Available online at http://linux-ipv6.org/.

[20] R. Bradshaw, N. Desai, T. Freeman, K. Keahey, “A Scalable Approach to Deploying and Managing Applicances”, Teragrid 2007.

[21] C. Sapuntzakis, et. al., “Virtual Appliances for Deploying and Maintaining Software”, LISA 2003.