2009-10-01

Desktop Virtualisation Scaling Experiments with VirtualBox

John Griffin
Dublin Institute of Technology, john.griffin4@student.dit.ie

Paul Doyle
Dublin Institute of Technology, paul.doyle@dit.ie

Desktop Virtualisation Scaling Experiments with VirtualBox

John Griffin [1], Paul Doyle [2]

[1] School of Computing, Dublin Institute of Technology, Ireland
John.griffin4@student.dit.ie

[2] School of Computing, Dublin Institute of Technology, Ireland
Paul.doyle@dit.ie

Abstract

With recent advances in virtualization and a growing concern regarding the administration and cooling costs associated with managing legacy servers, organisations are moving towards server and desktop virtualisation. Virtualisation provides the ability to provision a servers resources efficiently thus increasing hardware utilisation and reducing costs. While server virtualization provides clear advantages with regard to system management, higher system availability and lower recovery times, desktop virtualization is often complicated by the issue of determining the number of concurrent virtual desktops capable of running on a single server while providing acceptable performance to each desktop user. Determining the number of virtualised desktops capable of running on a virtual server is a non-trivial issue, and within most environments this is determined by trial and error. The objective of our experiments was to identify the maximum number of virtual desktop instances within our experimental environment. Each virtual desktop (guest operating system) was configured to automatically perform a specific tasks designed to strain the host servers resources to determine the breaking point of VirtualBox and to identify the maximum number of concurrent virtual desktops possible under 4 specific workloads.

Keywords: VirtualBox, Virtualisation, Sizing, Performance

1 Introduction

Virtualization technology has grown in popularity over the last decade. While its roots may be traced back to the 1970's with the original IBM publications on virtual machines [1] it is only relatively recently that it has been incorporated into corporate and educational computing infrastructures. Since VMWare [2] released the first virtualisation product "VMware Virtual Platform" February 1999 to address under utilization of servers, the number of virtualisation products available has dramatically increased with Xen [3], Virtual Server [4], and VirtualBox [5] being some of the most commercially successful examples. There are also many examples of virtualisation based enterprises being constructed such as Amazon's Elastic Compute Cloud (EC2) [6] and IBMs Cloud computing initiative [7]. Within education, virtualisation has been demonstrated as a key learning tool with virtual laboratories implemented by Border [8] and Buller [9]. There are two primary areas where virtualisation has become prevalent, server virtualisation and desktop virtualisation. Server virtualisation provides the ability to provide server based functionality within a virtual container allowing among other things, legacy systems to move from outdated hardware to more modern, higher performing systems, without the requirement for system reinstallation. Migrating from hardware installations to virtual installations is a service offered by most virtualisation technologies. Desktop virtualisation on the other hand provides the functionality of user desktops within a virtual

environment and typically involves the concurrent execution of virtual machine guest operating systems on the virtual host server. The focus of this paper is on desktop virtualisation. A series of 4 sizing experiments were designed to provide identical automated workloads within the virtual desktop guest operating systems and to slowly increase the number of concurrently running virtual desktops until a breaking point was reached. Each experiment used the same experimental procedure and setup with differing loads running for each experiment.

1.1 Structure of the paper

In section 2 we review virtualisation technology to provide an understanding of what the technology entails and review some of its uses today. Section 3 provides details of the experimental design and setup. Section 4 takes a critical look at the experimental data and section 5 provides conclusions and identifies future work. This paper is aimed at professionals within educational institutes seeking to understand some of the practical limits of virtualisation technology, and demonstrates an experimental method for use in future experiments.

1.2 Experimental aims and objectives

The aim of these experiments was to determine the upper limit on the number of concurrently running virtual desktops for the designed setup for each of the different loads. The loads used were – 1) No load running, 2) CPU Intensive loads, 3) Disk intensive loads, and 4) Audio intensive loads. To achieve these aims an experimental setup and procedure was required which is outlined in more detail in Section 4. The primary requirement for each experiment was to re-use the same implementation, and to have a shared set of terminating conditions. This allows a comparison between each of the experiments in Section 5.

2 Virtualisation Overview

There are currently many different platform virtualisation technology implementations in production. Each approach will undoubtedly provide varying results in a performance based experiment, however given the vast range of technologies available these experiments are limited to just once specific form of virtualisation, and one specific product, VirtualBox, which is emulation based virtualisation technology.

2.1 Emulation Virtualisation

Simulates complete operating system hardware inside the virtualisation server application. Instructions can take a longer to reach the native hardware due to the additional server software layer between the Guest OS and the underlying hardware. VirtualBox is an example of emulation virtualisation as shown in Figure 1. A Virtual Server application emulates all operating system hardware for the Virtual Machine Guest Operating systems. It should be noted that that there are other virtualisation methods such as paravirtualisation, and hardware virtualisation which are well described by Rose in his review of system virtualisation techniques [10]

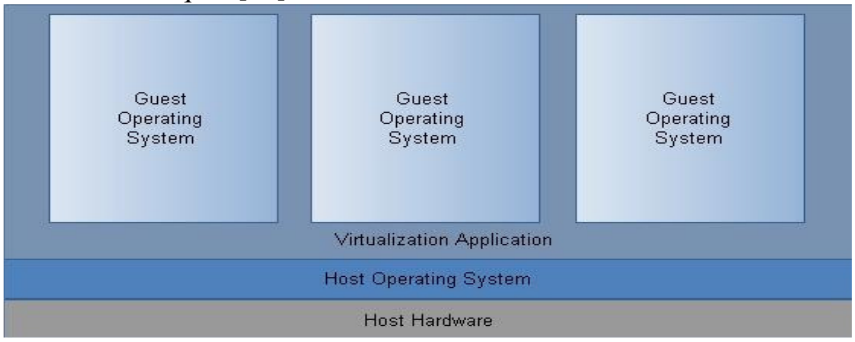

Figure 1. Emulation Virtualisation

2.2 VirtualBox

VirtualBox is an open source virtualisation software package designed by innotek and developed by Sun Microsystems. VirtualBox can be installed on host systems running Linux, Windows, Open Solaris and Mac OS X and can support a large variety of guest operating systems such Linux, Solaris, OpenBSD and most Windows desktop or server operating systems. VirtualBox has a large community backing and since its release on the 15th of January 2007 it has become a major player in the virtualization market. Given its popularity and the fact that it is open source, VirtualBox was chosen as the primary virtualisation technology for these experiments. *VRDP* and *VBox Manage* were also two key features which were essential to the running of these experiments.

2.2.1 VRDP

VirtualBox has a VRDP server (VirtualBox Remote Desktop Connection) built in that allows remote connection to any of the running virtual guests using any standard remote desktop program such as rdesktop on Linux and the built in RDP for Windows. Using this feature allows for direct control of the virtual machines from any computer on the network by using the host servers IP address and the port number it was assigned. This allows for the viewing of the guests output and the direct manipulation of the operating system as if it were running on the machine being used to remotely view the guest.

2.2.2 VBox Manage

VBox Manage is a command line tool that provides access to the array of features provided by VirtualBox that are not available from the GUI. This tool allowed for the automation of some of the controlling aspects of the experiment including cloning, running and stopping virtual guest machines.

3 System Implementation

There were 4 experiments designed to test the limits of the VirtualBox based virtualisation environment shown in Figure 2. The environment was constructed using the components shown in Table 1 and Table 2.

Table 1. Host Server Setup

Roles	Description
Server Hardware for Virtualisation	SunFire X4150, Dual Quad Core, 20GB RAM.
Server Operating System	Ubuntu 8.10
Server Virtualisation	VirtualBox 3.0.0
Server OS Monitoring	KSar 5.0.6
Ping Server	SunFire X4150 running Ubuntu 8.10
Remote Connection to Guest OS	Windows XP remote desktop

Table 2: Master Clone Setup

Roles	Description
Guest Operating System	Windows XP Service Pack 3, 256MB RAM
Guest OS Monitoring	MMC (Microsoft Management Console)
Load Execution Automation	Macro Recorder, Jbit 3.64.0
CPU Load	MaxxPI2 Preview
Disk Load	Bart's Stuff Test 5, version 5.1.4
Audio Load	WinAmp 5.56

Each virtual guest is cloned from a Master Clone. This clone was configured with unattended automation in mind, all load generators and audio files were in place and configured prior to cloning.

The primary function of this setup was to create a virtualised environment which could potentially support a large number of concurrently running virtual machines. Ubuntu 8.10 was chosen to be the Host OS as it provided access to all RAM due to the fact that it is a 64bit operating system. Windows XP was chosen as the Guest OS because it is the primary operating system within the Dublin Institute of Technology so it was considered the most relevant operating system for our target reader. The implementation tasks were broken down as follows.

a) Building a virtualisation server on the x4150 platform.
b) Creating an XP Service Pack 3 Guest OS.
c) Designing a series of loads to be executed by the Guest Operating System.
d) Configuring the Guest OS upon startup to automatically mount a shared network location via SAMBA then to retrieve a small load generator script.
This script tells the guest OS which of its stored load generation profiles to execute.
e) Configure each Guest OS to monitor its own system behaviour using MMC.
f) Configure kSar on the Host OS to record host statistics while Experiments are running.
g) Clone the Guest OS system 100 times so that there are a large number of identical Virtual Guests which can be started during the experiments.
h) Create a series of start and stop scripts for the Host OS to allow Guest OS clones to be started or stopped through the command line, depending on the experimental requirements.
i) Create a Ping Server which constantly pings each of the running Virtual Guest systems to show that they are still visible to the network during the experiments.
j) Access each of the running Virtual Guest systems using a windows RDP client to show that the virtual machines were available to the user. VirtualBox provides VRDP ports for each Guest OS, allowing full graphical connections to be made on any running guest with just the host servers IP address and its assigned port number.

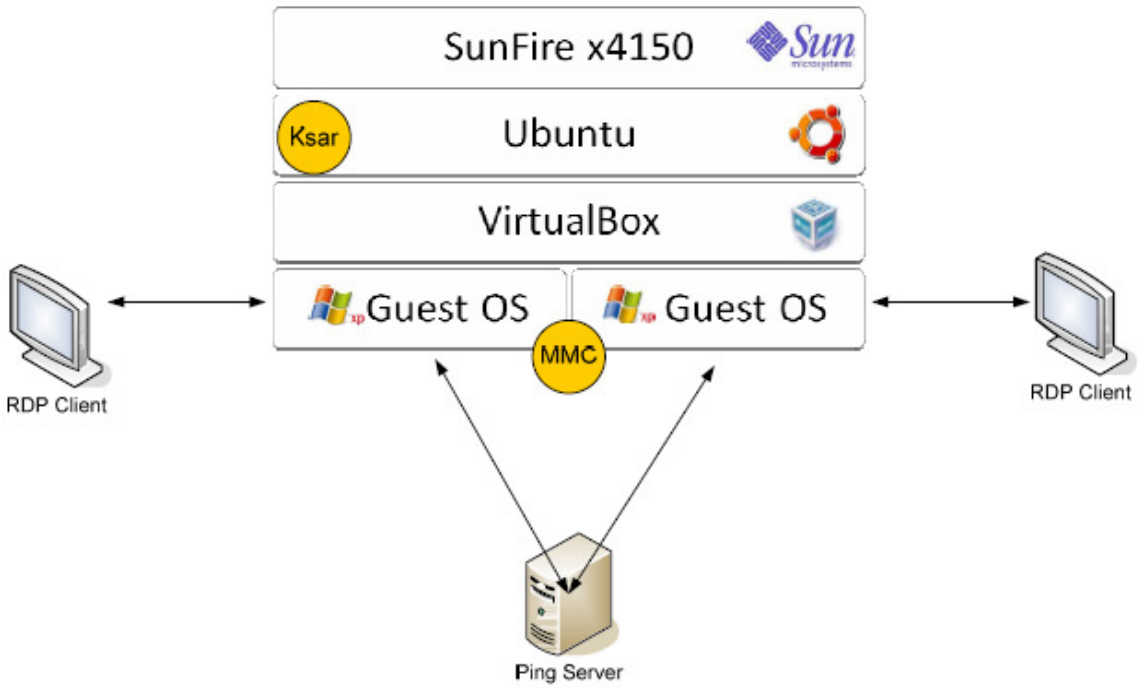

Figure 2. Experimental Setup

For the purpose of direct manipulation each Guest OS Virtual Machine was assigned both an IP address and a VRDP port number by the guest creation scripts. A series of scripts were created to allow the Clone VMs to be started from the command line of the Ubuntu server and these scripts were used as part of the experimental process. For each of the load generators an application was selected which stressed that specific aspect under test, and the Macro Recorder was used to automatically select and run the application once the clone VM was started.

4 Experimentation

There were 4 distinct sizing experiments run which all aimed to identify the maximum number of concurrently running Virtual Machine Clones running the same load before an experimental termination condition was reached. The experiments were numbered as shown below.
Experiment 1: Clone Virtual Machines started with no load generator application running.
Experiment 2: Clone Virtual Machines started running a CPU Intensive application (MaxxPI)
Experiment 3: Clone Virtual Machines started running a Disk intensive application (BS Test v5.1.4)
Experiment 4: Clone Virtual Machines started running a streaming audio file (WinAmp 5.56)

Each experiment followed the same experimental process with the primary difference between each being the load run by the VM Clone once it started running.

4.1 Experimentation Protocol

The following protocol was used for each experiment.
a) Power On the Virtual Server Hardware and verify that the VirtualBox was running.
b) Start the kSar monitoring software on the Host OS.
c) Start the Ping Server and record the ping responses to each of the possible VM Clones to be run.
d) Start the first 5 VM clones.
e) Check the termination conditions for the experiment.
f) Continue to start VM clones verifying that the termination conditions after each clone is started.

4.2 Experimentation Termination Conditions

To ensure that all experiments had a definitive end point a series of conditions were identified which marked the end of the experiment. Only one of the conditions needed to be met for the experiment to complete. The reasoning behind these controls was to identify realistic boundaries which would effectively cause a service outage to the user of the Virtual Desktop. The termination conditions were as follows.

a) *The Virtual Machines fail to respond to Pings from the Ping server for more than 10 seconds.* The purpose of this was to ensure that the virtual machines were visible on the network. Once they were not visible then this effectively was a service outage condition.

b) *The Virtual Machine failed to load the shared network point to retrieve the load generator.* The purpose of this condition was to ensure that the virtual machine could use the network to identify the correct load to run. If the VM could not identify the correct load then it effectively could not take part in the experiment. During the experiments any network traffic was minimal due to the fact that the load generator script was less than 1kb in size.

c) *The Virtual Machine allowed RDP connections to the GUI desktop over the VRDP assigned port.* The VM only offers a service to users once it is accessible to them. RDP is the only method of accessing the virtual desktop.

d) *The Virtual Machine can continue to run the assigned load.* This condition was more difficult to monitor that the previous conditions in that the load generators continued to function but were doing so at a reduced rate. It was only during the audio testing that an exception was made to determine what "reasonable" service meant.

4.3 Experiment 1 – No Load

The purpose of this experiment was to identify the maximum number of Virtual Machines required for the remaining experiments. This experiment effectively provided an upper bound on the number of VM (clones) which would be required in the remaining experiments on the grounds that when the VM was doing no specific work, it should be possible to have more running than when a load was is introduced. For this experiment the VM were started in sets of 5. Each of the 5 VMs were required to pass the termination conditions before the next set of 5 VMs were loaded. The experiment was run over a 1 hour period until all VMs suddenly failed the Ping Server condition.

All of the VMs stopped responding to pings within a few seconds, and once this occurred, RDP access to the Virtual Machines was lost, including access to the Ubuntu Host OS. The following graph identified the point at which the failure occurred. At the point of failure there were 56 Virtual Machines loaded and running within our experimental setup.

Experiment Duration: 15 Minutes
Initialisation interval: Clones Initialised in groups of 5. Clone 1-25 < 1 Minute, 26-54 < 2 mins.
Result: Concurrently Running VMs = 56
Failure: Termination conditions "a,b,c & d" as the host became unresponsive.
Analysis: The server started to "Thrash" when all RAM available was consumed and the number page activity increased dramatically. This result clearly suggests that the number of VM was bound by the available RAM on the Hardware Platform.

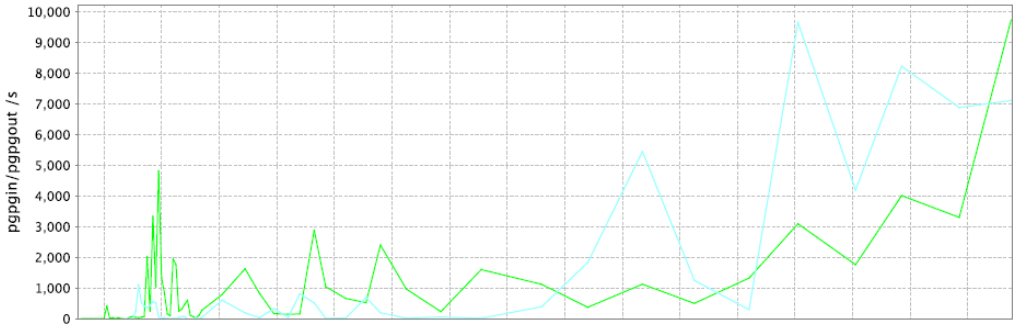

Figure 3. Host Paging Activity

4.4 Experiment 2 – CPU Load

The point of this experiment was to check how a CPU intensive application would impact the use of the VM machines. When each new VM was started it mounted the shared network mount point and copied the Macro Recorder script designed to run the MaxxPI2 [11] application. This software is specifically designed to instruct the CPU to calculate PI to a configurable precision. It was assumed that the CPU load would cause a significant lowering of the limit of concurrent VMs so they were started in batches of 1 initially. The experiment was run over approximately a 2 hour period. Despite initial issues with each of the system loads stalling and having to be restarted, the CPU Loaded VMs continued to pass the termination conditions until there was a failure to start the MaxxPI application when VM clone 49 was started.

Experiment Duration: 132 Minutes
Result: Concurrently Running VMs = 49
Initialisation interval: Clones 1-19 < 1 Minute, 20-41 < 2 Minutes, 42-49 <4 Minutes
Failure: Termination condition "d" failure to run the load generator script.
Analysis: The Host CPU ran at 100% utilisation early on in this experiment. However the Virtualisation software and the Host OS managed the CPU resource ensuring that while the CPU allocation to each process was dramatically reduced, the number of VMs running was much closer to the first experiment than was expected.

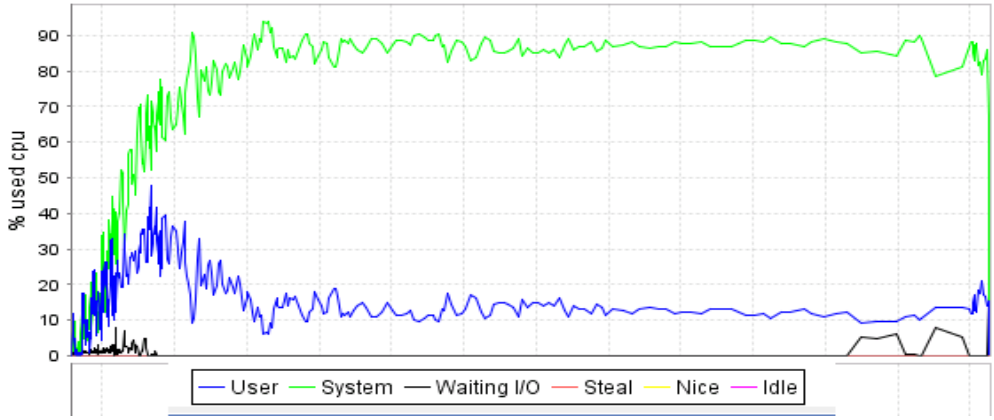

Figure 4. Host CPU

4.5 Experiment 3 – Disk Load

This experiment used the *Bart's Stuff Test 5, version 5.1.4 free edition* [12] a common win32 bit stress test for storage devices. The primary aim of this test was to emulate highly intensive disk applications. Since this application writes to what it considers the hard disk (which is actually only a file on the host operating system) it stresses the underlying Host OS virtualisation software. The VMs were added one by one until a termination condition was reached. In this experiment there was a failure to mount the shared network which is a precondition of obtaining the required system load to run within the VM. The ability to access the mount point was restored only after the number of VMs running was reduced.

Experiment Duration: 27 Minutes
Initialisation intervals: Clones 1-8 > 2-3 Minutes, Clone 9 > 3 Minutes
Result: Concurrently Running VMs = 11
Failure: Termination condition "b" failure to mount shared network point
Analysis: It is not clear what limitation was reached which failed to allow the network shared point to be accessed. Initially it was considered to be a potential issue with SAMBA, however each of the VMs when loaded individually managed to connect to the shared location, load their profile, then disconnect.
We also considered the possibility of a disk drive read/write limitation, however when a clone located on a separate internal disk was initialised the problem persisted. This functionality had not failed in prior experiments and the ability to mount shared stored was restored once there was a reduction in the number of VMs running.

4.6 Experiment 4 – Audio Load

This final experiment was the only one which required a qualitative analysis component. This analysis is an extension of the termination condition "d" where the VM is required to deliver service to the load applied to each VM. The termination condition was defined as the point at which the audio file could not play a sustained uninterrupted audio stream over RDP to the connecting user. Each of the VMs loaded an MP3 playlist using Winamp and then played each song in turn from audio files stored locally on the guests file system. Virtual machines were added one by one until the audio stream started to have regular drop in audio for approximately 0.5 of a second every 5 seconds. This behaviour was consistent across all of the running VMs. The quality of the audio stream improved once the number of VMs was decreased.

Experiment Duration: 14Minutes
Initialisation intervals: Guests 1-5 > 1 Minute, 6-35 < 2 Minutes
Result: Concurrently Running VMs = 35
Failure: Termination condition "d" failure to run load generator within qualitative boundaries.
Analysis: This experiment is the closest to a real world deployment where there is a qualitative component. There was a clear correlation between the number of VMs running and the frequency of interruptions to the audio stream. From the analysis of the Host logs it would appear there is a CPU based limitation.

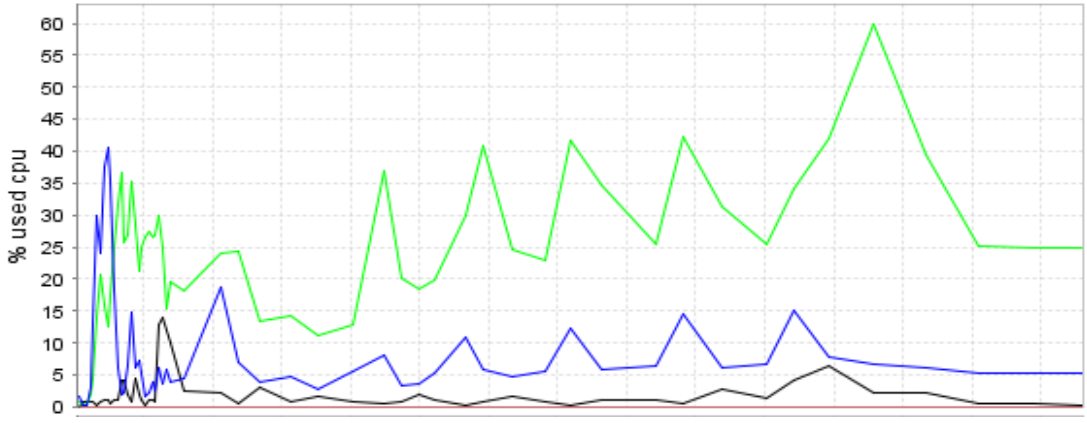

Figure 5. Host CPU Utilisation

The aim of these experiments was to identify some of the basic limitations with running virtualised desktops under VirtualBox. There was no effort made to optimize or tune the host or guest operating systems. Efforts were made to eliminate any network related issues by running the experiments in an isolated network environment. The times recorded by the Ping Server for example while indicating higher latency as the number of virtual machines was increased could be attributed to the increase in network activity associated with each of the running virtual machines, although this would seem unlikely. With the exception of the final experiment no qualitative analysis was performed on the performance of the running loads within the VM guests OS systems.

5 Conclusion and Future Work

While these experiments focus on VirtualBox it is however not possible to take those results and apply them to all VirtualBox installations due to the fact that VirtualBox runs on multiple host operating systems. It is quite conceivable that the limitations observed are underlying issues within the Host OS and not within VirtualBox itself. It is however possible to take these results and to build reference installations within the parameters of the experimental setup (specified hardware and Host OS configuration) where there is a definitive upper limit on the number of VMs which should be allowed to run. Obtaining the optimum number of VM desktops which would provide acceptable levels of response and performance would require further knowledge of the pattern of application use by users on that system. Further research planned to compare the behaviour of VirtualBox running on different Host OS environments providing further points of comparison for these experiments. As VirtualBox is an open source product it should be possible to build instrumented versions of the product to assist in identifying more specific causes for poor system performance. An experiment to test the quality of video output was conducted but termination condition "d" was reached when the first clone began to playback the local video file. This was due to Windows RDP being unable to display to 32 bit colour. Further experiments in video performance and more complex user load profiles is also planned to provide a more comprehensive understanding of scaling for real world deployments.

6 References

[1] R.P. Parmelee, T.I. Peterson, C.C. Tillman, and D.J. Hatfield, "Virtual storage and virtual machine concepts," *IBM Journal of Research and Development*, vol. 11, 1972, p. 99.

[2] "VMware: Virtualization via Hypervisor, Virtual Machine & Server Consolidation - VMware." Available: http://www.vmware.com/

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *Proceedings of the nineteenth ACM symposium on Operating systems principles*, Bolton Landing, NY, USA: ACM, 2003, pp. 164-177.

[4] "Microsoft Virtual Server 2005 R2.", Available: http://www.microsoft.com/

[5] "VirtualBox - VirtualBox.", Available: http://www.virtualbox.org/

[6] "Amazon Web Services Amazon.com.", Available: http://aws.amazon.com/ec2/

[7] "IBM Press room - 2007-11-15 IBM Introduces Ready-to-Use Cloud Computing – United States.", Available: http://www-03.ibm.com/press/us/en/pressrelease/22613.wss

[8] C. Border, "The development and deployment of a multi-user, remote access virtualization system for networking, security, and system administration classes," *Proceedings of the 38th SIGCSE technical symposium on Computer science education*, Covington, Kentucky, USA: ACM, 2007, pp. 576-580.

[9] J. William I. Bullers, S. Burd, and A.F. Seazzu, "Virtual machines - an idea whose time has returned: application to network, security, and database courses," *SIGCSE Bull.*, vol. 38, 2006, pp. 102-106.

[10] R. Rose, "Survey of system virtualization techniques.",Available http://hdl.handle.net/1957/9907

[11] M. Bicak, "MaxxPI²," 2009, Available: http://www.maxxpi.net/

[12] Bart Lagerweii, "Bart's Stuff Test 5," 2005, Available: http://www.nu2.nu/bst/