Analysis of the Usability of Xen with Mysql Databases

Richard Hansen

Office of Science, SULI Program

University of Chicago

Argonne National Laboratory

Argonne, Illinios

August 18, 2006

TITLE PAGE

ABSTRACT

Analysis of the Usability of Xen with Mysql Databases (RICHARD HANSEN, University of Chicago, Chicago, IL 60637) ALEXANDRE VANIACHINE (Argonne National Laboratory, Argonne, IL 60439).

The Xen Linux kernel is specially designed for emulation purposes. It can be built into a standard Linux build with relatively minor alteration of the kernel. It allows for the creation of virtual machines running Xen compatible operating systems with very little overhead. An important concern is how efficient and how stable are Xen VMs. It was determined that it was required to run tests on Xen VMs in conditions approximate to the real world, specifically database employment. A Xen server was built using and from there two identical VMs were created. A TAGS database was then replicated on all the systems. A variety of tests were run against the setup. These included loading data into a database, running select queries, and additionally running some specialized testing tools. The tests were run individually against each server, and also in tandem, for example running a writing test on one VM while simultaneously running a reading test on another. All tests results seem to indicate that Xen runs flawlessly and is perfectly suited to running a VM with a database. There were also some interesting results regarding how the system allocates resources that should be taken note of if the system was to be deployed.

INTRODUCTION

Under the current grid setup, if a person wishes to run a job requiring a special machine setup, considerable work needs to be done. The person would need to contact the administrator in charge of the grid node that he wishes to use and request that a machine be set up on the grid. Hopefully the machine is then set up properly. One of the most common situations in which this occurs is when a job requires a local database with a specialized setup.

Recently a new modification to the Linux kernel known as Xen was developed at the University of Cambridge [1]. Basically Xen allows the emulation of other operating system in completely independent environments. This provides a unique solution to the problem mentioned above. Rather than requesting a machine to be set up, one only needs to request access to a virtual machine. This virtual machine could conceivably be configured to run any type of environment the user wished, thus removing limitations along the lines of what type of systems the node administrator normally deals with.

Generally Xen is installed into a system running a standard Linux distribution, and it will boot up and run over the standard kernel. This machine is known as the host machine. A virtual machine (VM) can then be created on a properly set up system. Each machine may be refereed to as a domain. The host is always domain 0. The system is created by creating an image file which will become the ‘hard drive’ for the VM. A standard configuration file determines how many system resources will be allocated to this virtual machine.

Once an OS has been installed in the image file and the VM has been booted, it runs as if it was a completely independent system. A consol can be accessed through a

Xen tool, or an ssh connection can be established to the VM. The VM will actually have its own unique IP address and seem to be on the network as if it was a physically separate machine. The host machine will create a virtual router and the VM's will have virtual Ethernet cards. Granted all domains will need to share the bandwidth that is available to the host machine.

The Xen documentation claims that there is little overhead, and that a virtual machine will run with the performance of an actual machine with the same allocated system resources. In regards to physical memory, Xen requires only 32 megabytes to run the necessary software. This is supposedly the only major overhead.

However, there were concerns regarding how a Xen VM would perform under real world like conditions while running a database. The file backed method of creating an ISO to act as the VM's hard drive had also been reported as being unstable when under heavy I/O operations. The purpose of this experiment was to test Xen's performance in regards to database operations and ensure that it could be usable in actual applications.

MATERIALS AND METHODS

Our test computer, known as Atlas10, was running with a 2.6 Ghz Pentium 4 with 512 megabytes of RAM. The hardware of the machine used as a client machine in some of the tests is not too important as long as it is comparable to the test machine. The Client machine, known as Atlas12, was identical with the test machine with the exception of 1024 megabytes of RAM. The network is capable of data speeds of up to 100 kb/s.

When creating a Xen VM it is necessary to deviate from the standard procedure the Xen manual gives [1]. In our particular instance we had Xen running on a Fedora Core 4 system and created similar VMs. However, following the manual to create a VM would result in a VM that would not be able to boot. As such we followed the instructions listed on the Fedora Core Project Wiki [2], under the building a Fedora Core guest system section. It should be noted here that we used eight gigabyte iso files for the virtual machines. After this was done the ifconfig and route commands were used to configure the system for the network. Some additional configuration changes were needed, including the need to acquire some missing modules. Once one VM was created it was sufficient to simply copy its disk image file and configuration file and change the name. Once this was completed we had two VMs running Fedora Core 4 with mysql databases network ready. The memory was divided evenly among the domains with each virtual domain having 145 megabytes. The two virtual machines were given the designation Atlas13 and Atlas14

To prepare the mysql TAG databases that were used in the test it was sufficient to simply mount the virtual file systems and copy a preprepared database from the host system. Some additional library files were also needed for mysql to run properly but it was not a problem.

Essentially two different test types were preformed. One type used tools external from mysql to directly test the performance of the system. The other type would run various mysql jobs and record the time required for the job to complete.

The external tools that were used were IOzone [3] and iperf [4]. The first of these tools, IOzone, is to test disk and cache performance. It is an I/O test program. The other application, iperf, is designed to test network performance. IOzone was run on the host machine, on a VM, then on two VMs simultaneously. IOzone has a vast variety of tests it can run, however, for our needs we went with the random read and random write tests. These test will non-sequentially write and read to files of varriing sizes. To run iperf one machine must be set as a client the other must be set as a server where the client sends the server data. The various tests that were run included having Atlas12 act as the client and just the host domain on Atlas10, a VM, server, the host domain and one of the VMs, both VMs, and the host domain and both VMs acting as a server. It also included a test in which one of the domains was the server and another was the client.

The mysql tests were then divided into two categories, write and read tests. The writing tests involved loading TAG data into the database. The TAG data was from simulations as to resemble real world data that would be used in an actual application. It was 2.5 gigabytes in size. The read tests were performed by running select queries on the data that had been loaded into the database. The queries resembled real world queries and were sufficiently complex that they took a satisfactory amount of time to run. These tests were run from the client machine with the Xen domains acting as hosts.

Like the previous tests, the mysql tests were run in a variety of configurations. They were run on each domain independently to get some base data. They were then run in combinations. The different combinations were that connections were made to both domain 0 and one of the VMs, connections were made to both VMs at the same time, and a read test would be run on one of the virtual machines, while a write test was being run on the other one.

RESULTS

An interesting observation will be made before going into the results in detail. This observation is in regards to the host domain. Several of the tests, such as the database loading test, would cause the host machine to run flat out. The CPU usage would hover around 97%. When a situation like this would arise, in which there is much work for the host machine, the VMs available CPU power would drop down to less than one percent. As such it was impossible to run most tests in parallel between the host machine and a virtual machine as the VMs would be reduced to a crawl. It can also be noted that in the iperf test Atlas12's CPU usage never exceeded 5%.

First we will discuss the results of the IOzone tests. The peak of each 'mountain' is affected by the CPU cache while the area to the right of it is affected by the buffer cache. Comparing figures 1 and 3 they seem fairly similar. The differences in performance are that in 3, the peak is missing and the area affected by the buffer is much smaller. Figures 2 and 4 are also similar. Comparing figures 3 and 4 with figures 5 and 6 respectively it can be seen that the performances are comparable, but the buffer area took a large hit during the dual run test.

Looking at the iperf results in table 1, it can be seen that connecting with Atlas12 with any given domain returns results just under the hardware limit as could have been expected. Communication between two VMs actually exceeds this, while communication between a VM and the host domain more than doubles this. Also to note is that when multiple domains are acting as servers, the total data speed adds up to approximately the hardware limit. It also seems that the host domain uses twice the amount of bandwidth that it allocates the VMs. The VMs seem to share resources evenly.

Looking at table 2, it seems that the performance of the host machine is slightly better in write test; however the VM does not do too poorly. When two VMs are being written two the time to completion is approximately double. In table 3 the reading tests show similar behavior. The results of table 4 are actually better than what would be expected from simply doubling the times from the previous tests.

All the results on tables 1-4 are rounded averages from a series of runs. At no time during the tests where errors reported.

DISCUSSION AND CONCLUSION

As mentioned it seems that when the host domain is under heavy load it takes resources from the VMs until the task is done. This is important to keep in mind because it means that the host domain should be kept free of any intensive applications if one wishes all of the VMs to continue running smoothly. However, in a situation where the host domain is using the majority of the process power, the VMs still continue to run just extremely slow. There does not seem to be any issue with the VMs sharing CPU time with one another.

The IOzone tests seem to indicate that the VMs are unable to benefit from the CPU’s cache. This is most likely reserved for the host, or simply occupied with Xen activities. The VMs also appear unable to access as much as the buffer. However, other than that the results were reasonable. It could be interesting to see how this behavior changes in a multi CPU environment.

The iperf results were encouraging. Once again they demonstrate Xen’s trend of giving more resources to the host while it is occupied. However, as long as the host is not being given jobs this should not be a problem.

The mysql results, perhaps the most important, are very encouraging. A single VM appears to be able to run just as well as an actual machine. There also appears to be little performance loss while increasing the number of VMs.

Also heartening is that at no time was there an indication that the file backed system was a problem. This was an initial concern due to reports that it would be unstable while under heavy I/O load.

The tests that have run seem to indicate that Xen would do very well as a database solution. While there are a number of other tests that could be run, there are a few in particular that seem like they may be useful to perform. It could be helpful to see how Xen performed in a multi core or multi CPU environment. It might also be useful to check how the replication and creation of new VMs affects ones that are already running. This is a fairly intensive process for the host machine so it could potentially bring the system to a crawl.

In the tests we ran there were two bottle necks. In the iperf tests it was bandwidth, while in the database tests it was CPU power. It would be interesting to rerun these tests if these constraints were improved.

Varying the OS’s run on the VMs would be an interesting and realativly easy task. One could then see if Xen deals with each OS. However reference systems would be needed. More complicated but potentially more rewarding would be varying the host domain’s core OS.

Finally it might be prudent to see how Xen handles file backed systems of differing sizes. It is possible that complications might arise with files larger than 8 gigs.

ACKNOWLEDGMENTS

This research was performed at Argonne National Laboratory. I would like to thank the U.S. Department of Energy for allowing me to participate in this internship program and have a chance to perform this work. Thanks must also go to my mentor Alexandre Vaniachine and to Edward May.

REFRENCES

[1] Ian Pratt, "The Xen virtual machine monitor," University of Cambridge Computer Laboratory. 2006, http://www.cl.cam.ac.uk/Research/SRG/netos/xen/documentation.html

[2] Thomas Chung, "FedoraXenQuickstart" in Fedora Project Wiki. Febuary 2006, http://www.fedoraproject.org/wiki/FedoraXenQuickstart

[3] "Iozone Filesystem Benchmark," Great Lakes Academy. May 1006, http://www.iozone.org/

[4] "Iperf 1.7.0 - The TCP/UDP Bandwith Measurment Tool," NLANR/DAST. May 2005, http://dast.nlanr.net/Projects/Iperf/

Figure 1

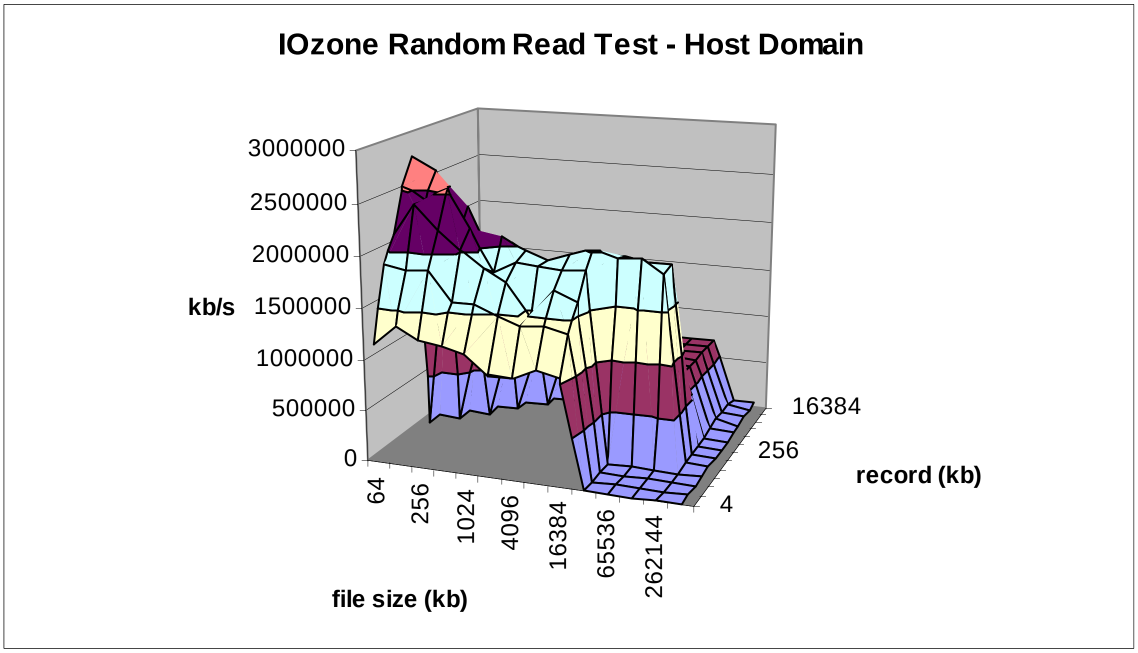

Figure 2

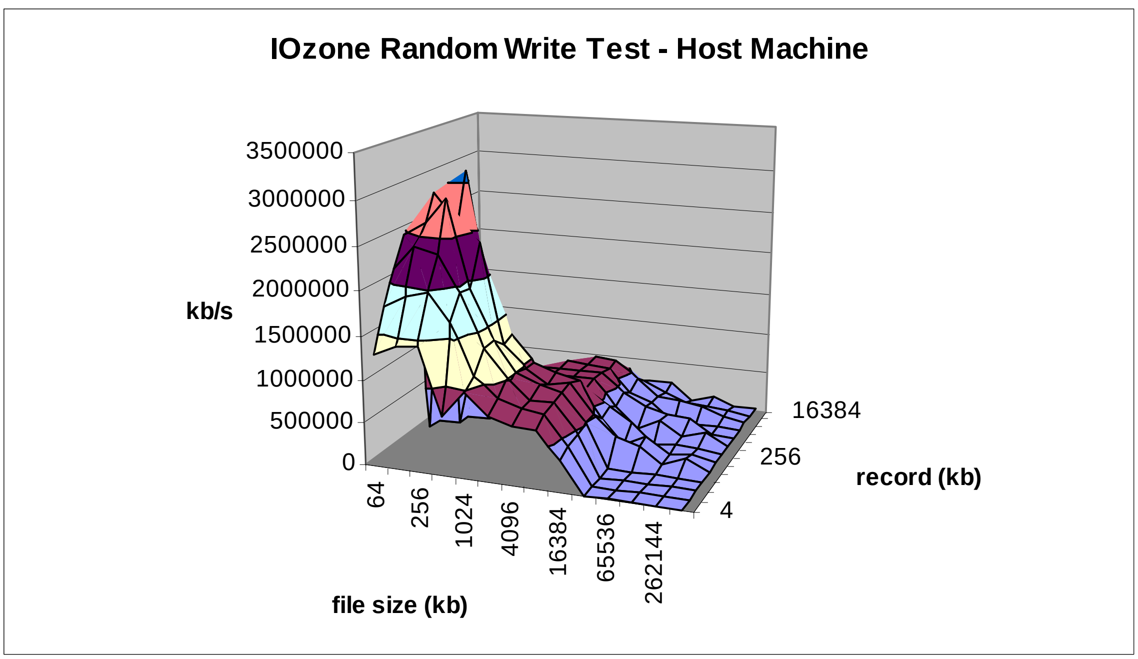

Figure 3

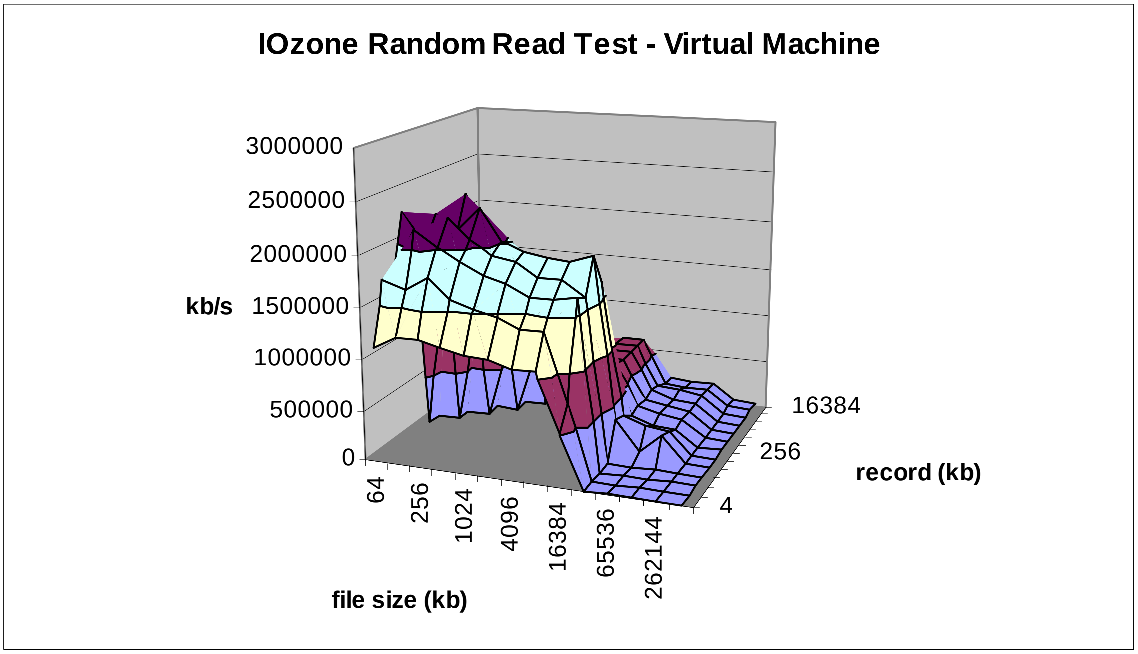

Figure 4

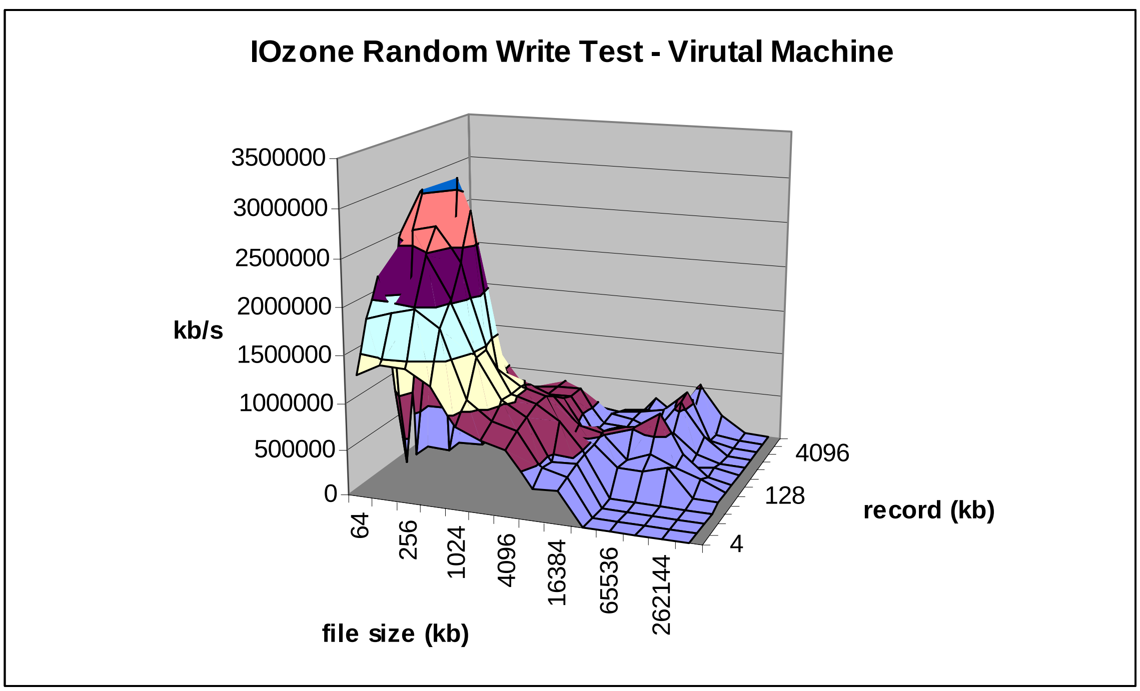

Figure 5

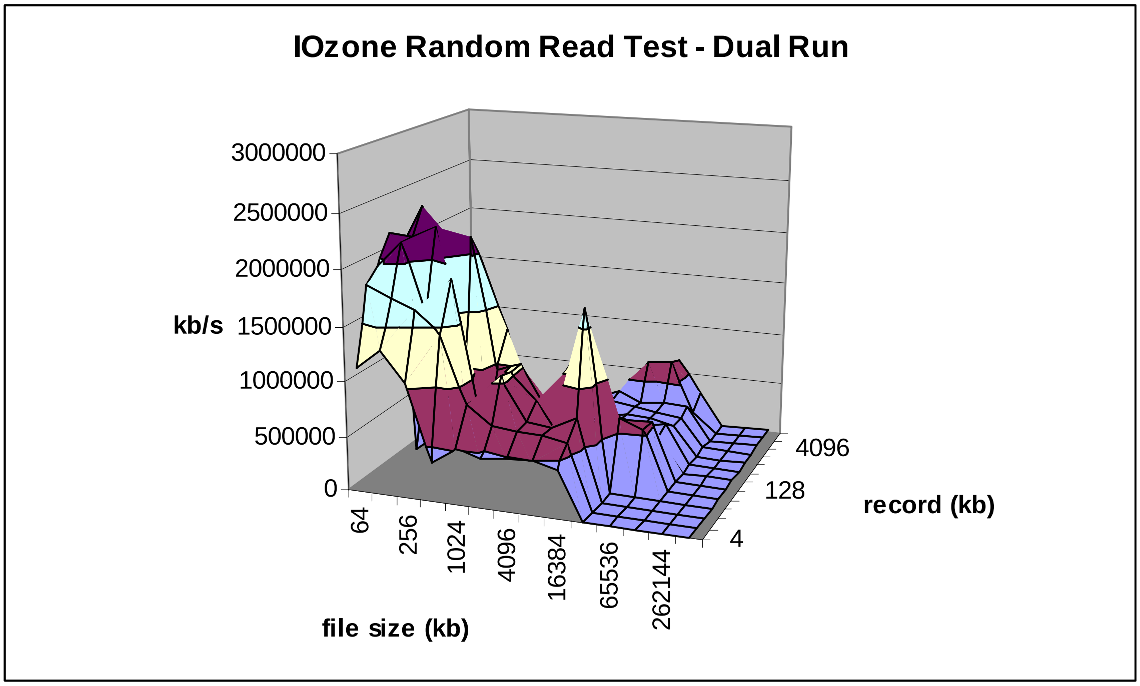

Figure 6

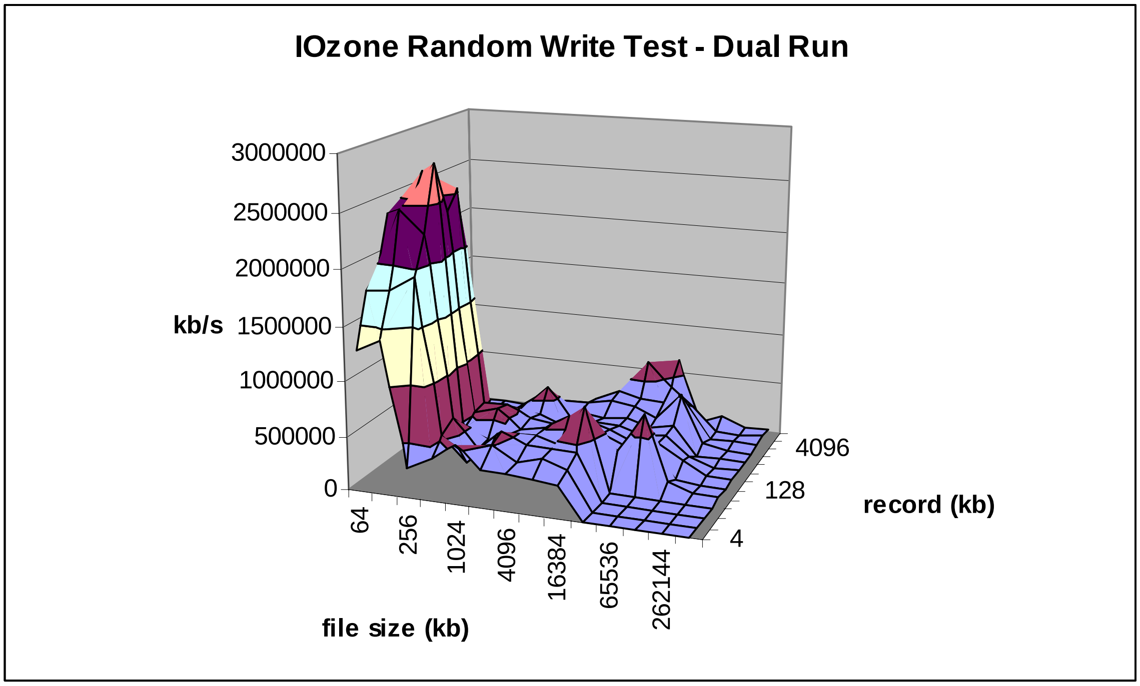

Table 1

ifperf tests

Server	Client	kb/s
VM	Atlas12	94
Atlas12	VM	94.1
Atlas12	Domain 0	94.1
Domain 0	Atlas 12	94.2
VM	VM	135
VM	Domain 0	230
2 VMs	Atlas 12	47 / 47
Domain 0 + VM	Atlas 12	62.4 / 31.4
Domain 0 + 2 VMs	Atlas 12	46.8 / 23.7 / 23.7

Table 2

Mysql Database Writing Test

Domain	Time (sec)
Host	416
VM	542
2 VMs	1142 (each)

Table 3

Mysql Database Reading Test

Domain	Time (Sec)
Host	391
VM	477
2 VMs	1094 (each)

Table 4

Mysql Read / Write Test

Read Time (sec):	918
Write Time (sec):	920