



DISK IO BENCHMARK TEST RESULTS

FOR JOYENT
Revision 9

October 13th, 2010

Scope:

This report summarizes Disk IO benchmark testing performed in September of 2010.

References:

[1]: <http://blog.cloudharmony.com/2010/06/disk-io-benchmarking-in-cloud.html>

[2]: Svn repository: <https://svn.codespaces.com/ims/joyent-benchmarks>
username: joyent **password:** joyent

[3]: Raw test data: DISK-IO_Final_Results.xlsx

[4]: Phoronix Test Suite 2.6.1:
<http://www.phoronix-test-suite.com/download.php?file=phoronix-test-suite-2.6.1>

[5]: http://www.iozone.org/src/current/iozone3_347.tar

Joyent Disk IO Benchmark Testing Report

Introduction

The Disk IO testing was performed as part of a larger benchmark effort intended to provide a basis for comparison between Joyent SmartMachines and other virtual servers offered by cloud service providers.

Earlier in 2010, CloudHarmony engaged in an extensive benchmarking effort intended to provide "information and analysis to enable educated decisions pertaining the adoption of, and migration to cloud services". Their results and analysis are presented in a series of articles published online ref[1]. The CloudHarmony blog does not contain results for Joyent SmartMachines. Our testing procedures are intended to follow CloudHarmony's efforts as closely as possible and extend benchmarking for Joyent SmartMachines.

Instead of trying to reproduce all of the CloudHarmony results, we focused on those outlined for the Amazon EC2 servers used in their benchmark tests ref[1]. Our tests closely approximate the methods from CloudHarmony in regards to calculations and tests used. Figures for the Joyent SmartMachines should be a useful addition to the other benchmarks included in CloudHarmony's blog. It should be noted that not all test executables and versions contained in this report are identical to those of CloudHarmony due to differences in operating systems. These results should not be compared side-by-side to those of CloudHarmony. Our mathematical baselines and server instances are however identical.

CloudHarmony standardized on CentOS 64bit as the operating system used for test servers except where it was unavailable. Joyent SmartMachines run an OpenSolaris based operating system. Due to the original tests being Linux based, modifications were necessary to execute similar binaries on the Solaris based system.

SmartMachines provide a "bursting" capability enabled by their OS that allows a service to use more processor and memory resources on a temporary basis than the guaranteed minimum. This differs from nearly all other cloud providers that

provide a fixed processor/memory configuration. While bursting capability can be a tremendous advantage to an operational system, it can complicate benchmark testing by stressing the system to its maximum capacity. On the Joyent SmartMachines the bursting capability allows a process on even the smallest server to potentially use nearly the entire compute and memory capability of the underlying hardware.

Joyent uses large commodity servers with 8 hyper-threaded processors that effectively yield 16 processor cores. This means that Joyent's smallest server, the 1 GB SmartMachine, can in some cases outperform Amazon's largest EC2 instance, m2.4xLarge. Due to this bursting capability, side-by-side comparisons may not be identical in nature between Joyent and other cloud providers.

Benchmark Setup

Storm on Demand 's Bare Metal Cloud Server was used as the baseline for all Disk IO tests. We compared both Joyent and Amazon servers to the Storm on Demand server baseline. The EC2 servers used consist of: m1.small, c1.medium, m1.large, m1.xlarge, m2.xlarge, c1.xlarge, m2.2xlarge, m2.4xlarge. The Storm on Demand and Amazon servers – 8 servers in 4 regions, were configured identically in terms of OS, CentOS 5.4 64-bit (or 32-bit in the case of EC2 m1.small and c1.medium where 64-bit is not supported). Joyent servers included: 1GB, 2GB, 4GB, 8GB, 16GB, 32GB.

All SmartMachines come with Joyent's SmartOS based on OpenSolaris - SunOS 5.11 snv_121. On the Joyent servers, the default gcc compiler is version 3.4.3 and the Amazon instances come equipped with gcc version 4.1.2. To provide comparison with an out-of-box install, the default software versions were used.

To run the majority of benchmark tests, CloudHarmony made use of the Phoronix Test Suite ref[1]. Version 2.6.1 was used for Joyent's OpenSolaris compatibility. There are several differences between version 2.2.0 used by CloudHarmony and 2.6.1 used in this report. These include test versions, source code, and executables.

The Phoronix Test Suite open source framework was originally a Linux specific tool and streamlines the testing procedures. Many of the included tests run natively under Linux, but not on Solaris as noted in the Phoronix documentation. Since the test suite makes use of shell scripts to download, unpack, build, install, and run the various tests, some modifications were necessary to run identical tests on the Joyent SmartMachines. Scripts within the Phoronix Test Suite make use of the Bourne Shell by specifying the `#!/bin/sh` command interpreter. Normally including `#!/bin/sh` works fine on most Linux systems, but under Solaris this is not the case.

Making the minor change from `/bin/sh` to `/bin/ksh` on Solaris can be made easily with a global search and replace to make all scripts compatible. Some other modifications required manual inspection and correction – versions and packages of required software are different between Solaris and Linux. There were also path changes needed in the test suite code to link to the appropriate executables in the Joyent SmartMachine OS. All changes to the test suite have been configuration controlled and is available to Joyent via subversion ref[2].

Benchmark Tests

Since the Joyent SmartMachine OS is OpenSolaris based, some tests that rely on Linux specific drivers or executables do not run correctly on a Joyent SmartMachine. The `hdparm`, `dbench` and `fiio` benchmark utilities fall into this category. Our testing results are generated from 4 of the 7 tests used by CloudHarmony.

Excluded: `hdparm` buffered disk reads, `fiio` and `dbench`

The following 4 tests were run and on both the Storm on Demand "Bare-Metal" server, Amazon EC2 instances, and the Joyent SmartMachines:

Included: `blogbench`, `bonnie++ v1.96`, `IOzone`, and `threaded IO tester`

As noted, the benchmark tests on Joyent required script customizations or source code modifications: [ref\[2\]](#) `phoronix-test-suite-2.6.1-SolarisPatched.tgz`. The benchmark executables test similar functionality on Joyent to the tests CloudHarmony ran on other servers. Calculations from the test results to derive our baselines are of an identical nature between the different Operating Systems.

Testing Procedures

The Phoronix Test Suite 2.6.1 was setup on Storm on Demand's new Bare Metal Cloud Server, Joyent and Amazon server to run all the Disk IO benchmarks except `bonnie++`. Phoronix compiles their results in xml files to be displayed in a web browser. The suite also creates image graphs for visual comparison. `Bonnie++` was run independently from the others with output saved to a flat-file for record keeping.

In order to reproduce our testing procedures on the Joyent SmartMachines see [ref\[2\]](#): `joyent_tests_directory.tgz`, `phoronix-user-directory.tgz`, `phoronix-suite-2.6.1-solaris.patch`. The following guidelines should produce similar or identical test results:

1. Install the Phoronix Test Suite into a local directory within the user's folder on each server. Tar files for Joyent are included [ref\[2\]](#): `joyent_tests_directory.tgz`, `phoronix-user-directory.tgz`. These two files contain all tests and executables that run on Joyent. If using these tar files, extract them into the user directory and skip to step 6.
2. If installing the default Phoronix Test Suite 2.6.1 [ref\[4\]](#), apply the patch file [ref\[2\]](#) `phoronix-suite-2.6.1-solaris.patch` to the test suite. This patch makes changes to the installation files and performs all alterations necessary for the Solaris platform. The test suite should be installed to a local user directory under `'Tests/phoronix-test-suite'`.
3. Install the tests via Phoronix. Double check the logs and especially `IOzone`. `IOzone` should have failed its installation. If the tests installed proceed to step 5. `Bonnie++` must be manually compiled and run if not using the supplied tarballs [ref\[2\]](#).

4. For IOzone, extract the ref[5] tar file into the IOzone created folder within the .phoronix-test-suite/installed-tests/ directory. Within the iozone3_347/src/current directory comment out the following lines to remove pit_server which is incompatible with Solaris.

```
makefile >> Lines 110, 368, 378.
```

Compile IOzone with the following command in the src/current directory:

```
gmake Solaris10gcc-64
```

Create an executable within the .phoronix-test-suite/installed-tests/iozone directory named iozone with the following code:

```
#!/bin/ksh
iozone3_347/src/current/iozone $@ > $LOG_FILE 2>&1
echo $? > ~/test-exit-status
```

Copy a pts_install.xml file into the iozone test directory and do a test run with the command ./phoronix-test-suite run iozone in the main Phoronix Test Suite install directory.

5. For the Joyent 1GB machine the following command needs to be modified in the setup script. Change the following line at 144:

```
-- java -Xmx2048m -d64 -jar SPECjvm2008.jar -ikv
++ java -Xmx2048m -d64 -jar SPECjvm2008.jar -ikv -bt 8
```

6. Use the setup_joyent.sh ref[2] script to execute the tests for Language. Be sure to change the last line to your own ftp server.

Note: If tests fail to run, make the following modifications to the test suite core files to see the full executable outputs for troubleshooting:

```
phoronix-test-suite/pts-core/library/pts-functions_shell.php
```

at line 110 add:

```
echo pts_variables_export_string($extra_vars) . "\n\n";
echo $exec . "\n\n";
```

This will output the Phoronix variables and executable to the command line.

Baselines

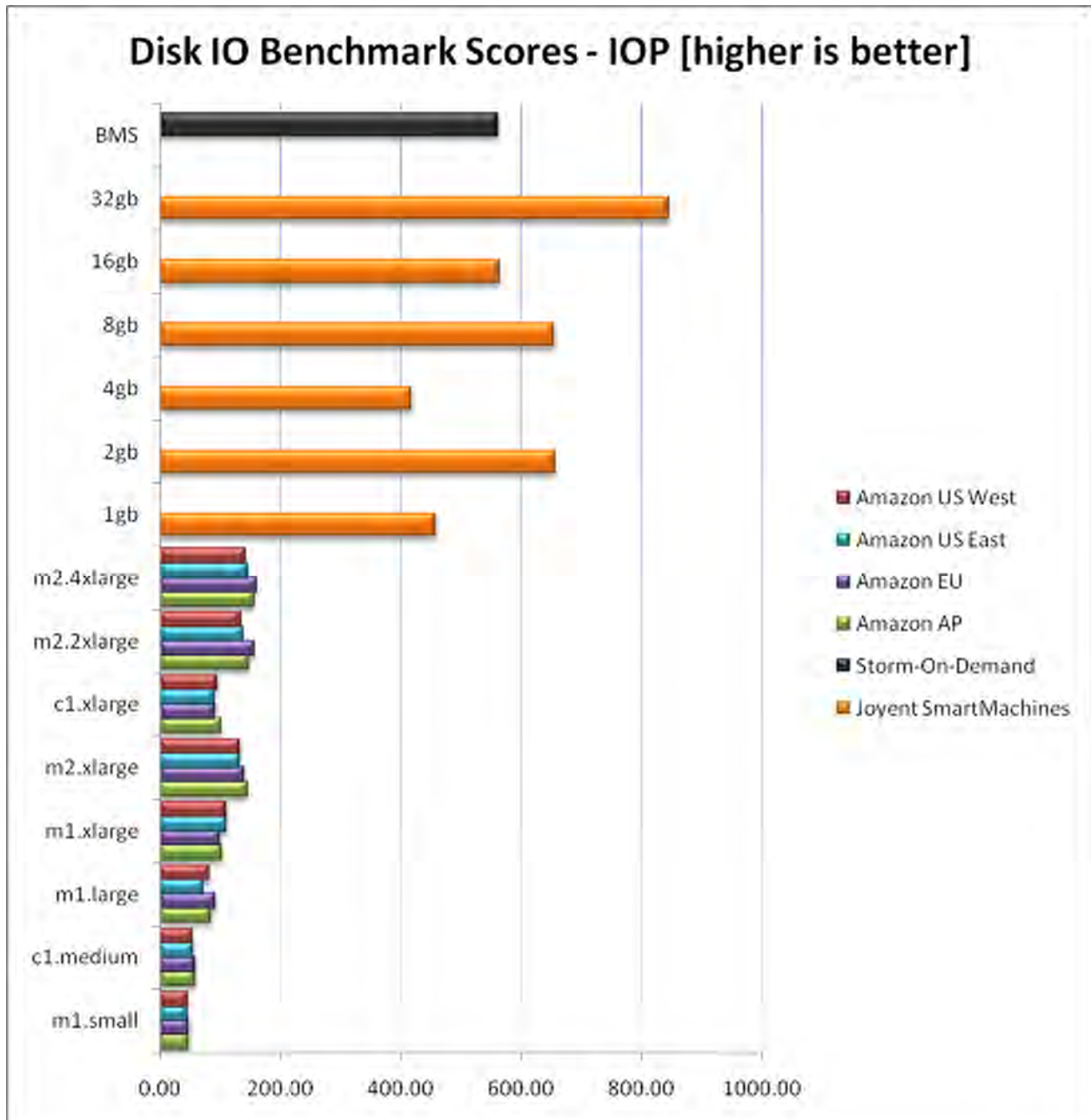
A baseline was taken from results of the Storm on Demand's new Bare Metal Cloud Server and calculated based on the methodology used by CloudHarmony. It should be noted that while only 4 benchmarks were used to calculate the IOP values in this test, the IOP results for the Amazon baseline servers were comparable to the corresponding results posted on the CloudHarmony blog.

Our calculations are the same used by CloudHarmony, but exclude the three tests which required Linux specific functionality: *hdparm*, *fiio* and *dbench*.

Test Results

For the full raw test data see ref[3]. Invalid tests have been excluded. The second set of Joyent data is raw data from the Phoronix Test Suite. Our IO performance score (IOP) 2 calculates 4 tests, IOP uses all 7.

To calculate IOP 2 please refer to the ref[3] DISK-IO_Final_Results.xlsx document. The IOP scores are compared and calculated against the Storm on Demand Bare Metal server tests. Our calculations are based on and have been verified against those found on the CloudHarmony blog.



	BMS	1GB	2GB	4GB	8GB	16GB	32GB
Storm on Demand	562.00						
Joyent SmartMachines		458.98	656.43	418.37	655.91	564.57	845.23
	m1.small	c1.medium	m1.large	m1.xlarge			
Amazon AP	47.15	57.98	83.88	102.86			
Amazon EU	47.43	59.67	92.66	99.02			
Amazon US East	46.33	55.49	72.39	110.67			
Amazon US West	47.50	55.11	83.60	111.93			
	m1.xlarge	m2.xlarge	c1.xlarge	m2.2xlarge	m2.4xlarge		
Amazon AP	102.86	145.32	100.48	147.33	157.76		
Amazon EU	99.02	140.82	92.02	156.81	160.29		
Amazon US East	110.67	132.67	89.44	138.19	145.78		
Amazon US West	111.93	133.92	96.76	134.50	143.11		

As visually shown, Joyent outperforms Amazon significantly, which can be a result of the Joyent architecture utilizing the OpenSolaris' ZFS file system. The Storm on Demand baseline is comparable to an average of Joyent servers.

Since the majority of our tests are compiled using Gcc, we needed to compare the impact of each operating system's default version. The Gcc version packaged with the Joyent SmartOS is 3.4.3, while the default Amazon CentOS version is 4.1.2. We compiled and ran our Tscp benchmark spot tests with different versions of Gcc and recorded the results. The comparison test scores between Gcc 3.4.3 and 4.2.3 show an approximate 14% increase in performance on both Joyent and Amazon EC2. Gcc 4.1 and 4.2 showed no difference in the results. Should the default Gcc version be upgraded to 4.x, the Joyent SmartMachine scores should be significantly higher.

Conclusion

From our calculations and test results, Joyent outperforms the established baselines across the Amazon instance sizes in terms of Disk IO performance. Compared to the Storm on Demand baseline, Joyent is comparable with 4 of 6 servers performing better. The high level of performance compared to Amazon and Storm on Demand, could be the result of the OpenSolaris ZFS file system.

The most equivalent comparison among the results is Joyent's largest server, the 32 GB SmartMachine and Amazon's EC2 m2.4xlarge instance. Both cloud servers have the same effective 16 processor cores and a comparable amount of memory. While analyzing these two similar servers, the Joyent SmartMachine easily outperformed the average of Amazon instances.

While comparing the Joyent SmartMachine IOP values, the 1GB to 32GB server sizes show little variation in Disk IO performance with an increase in server capacity. The variance in 1GB and 4GB SmartMachines could be analyzed further by running multiple tests and take the average. Occasionally, the Phoronix Test Suite produced lower than average scores, but after re-running the identical tests, scores were more appropriate. This anomaly within the Phoronix Test Suite could be a result of a multitude of possibilities. The Amazon servers as expected showed a slight consistent increase in the IOP values from the test results small to large based on server capacity.