Using load testing to find the bandwidth bottlenecks on two IaaS cloud providers.

By Alon Girmonsky, Performance Expert LTD, and Peter HJ van Eijk, Digital Infrastructures BV

Cloud computing is an emerging trend which enables the rapid deployment of new web based applications. While most cloud computing providers do not commit to a performance level or end-user experience, they encourage their users to test and benchmark before deploying.

When you host a server at your own facility, you know according to your network interface and the bandwidth provided by your ISP what to expect in terms of bandwidth allocation. If you find yourself asking what your bandwidth could be via a cloud provider, then this article is for you.

The goal of this article is to understand if there are any bandwidth limitations associated with two Infrastructure as a Service (IaaS) cloud providers, AWS and Rackspace, and if there are such limitations that can lead to bottlenecks, what they are.

Our approach was the following. We set up a simple web server on a simple operating system with the two cloud providers. Each of these was load tested to the maximum using a load testing service. (If you consider running these tests like these yourself, make sure to check with your cloud vendor prior to running tests to make sure that they are allowed and that all vendor recommended optimizations are in place.)

Choosing the cloud resource

The cloud resources used are listed in the following table.

	Amazon AWS	**Rackspace**
Type	Micro	512 MB
Memory	613 Megabyte	512 Megabyte
OS	Ubuntu 10.10	Ubuntu 10.10
Webserver	Apache 2	Apache 2

In order to be comparable we used a default configuration as much as possible.
The systems under test were located in the U.S.; the load testing service was located in Europe.

To benchmark the two providers, we chose a very basic load scenario. The load scenario simulates users executing a simple business process, which contains a short user sequence. In our scenario a user opens a browser, downloads a large file (3 or 4 Megabyte), waits for 10 seconds then loads a small HTML file (25KB) and then stops. The load scenario includes a gradual user ramp up from 0 to up to 100 concurrent users. This gradual ramp up should tell us if there is a bottleneck and where.

Results for AWS

The results are presented in the following graphs. It appears that the AWS resource could sustain a load of up to about 50 concurrent users before it got affected by the load. As is indicated in the next figure, above a certain amount of concurrent user sessions, response times start to increase. We call this the **load sensitivity point**.

The orange horizontal line indicates the maximum number of simulated users that can be active simultaneously without suffering from increased response times. (In this particular experiment we notice a small bump in response times, and a ragged performance at saturation. These patterns differ from experiment to experiment, and we attribute these to fluctuating traffic patterns and similar resource contentions).

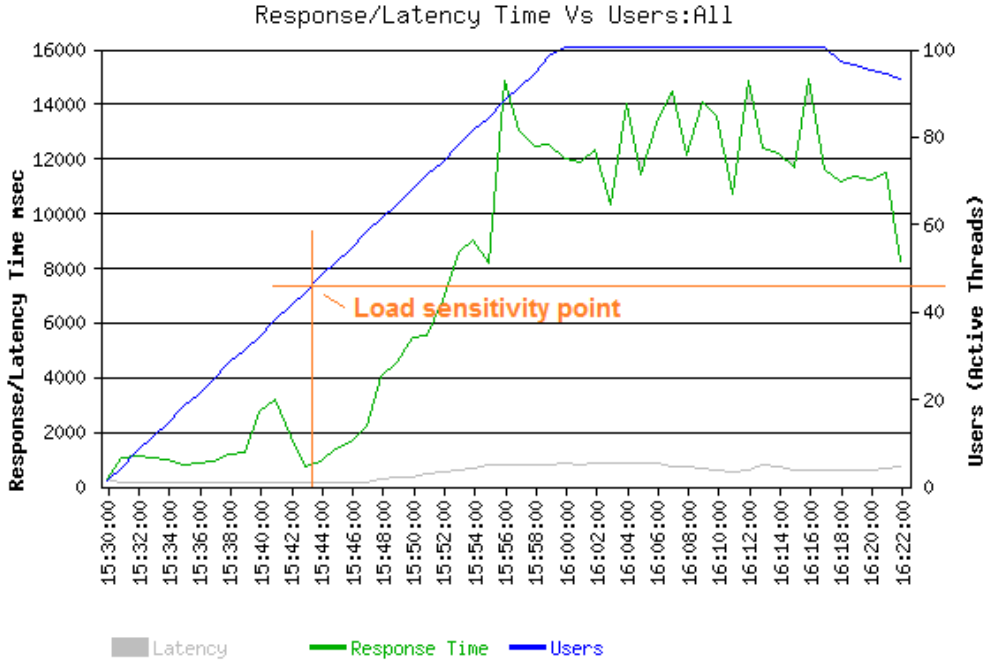

Figure 1 AWS instance 4 MB file – Response Time vs Simultaneous Users

What was the reason the load had this effect on the server? The next figure (figure 2) shows how response time increases when the bandwidth saturates. In this case, it appears that the bandwidth provided by this AWS instance was approximately 530 Megabyte per minute which is 71 Megabit/second.

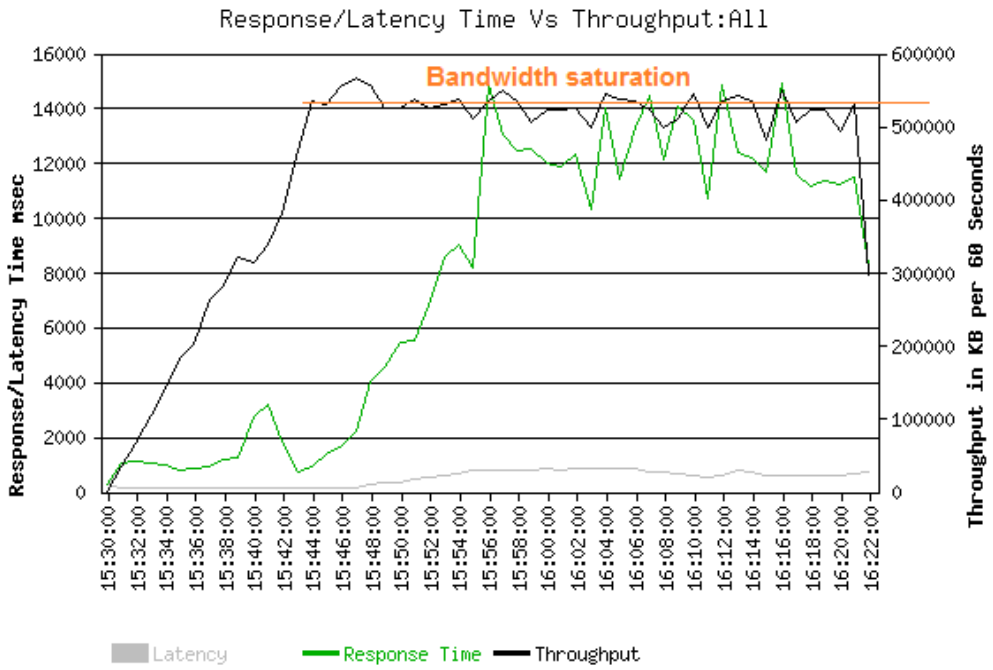

Figure 2 AWS instance 4 MB file– Response Time vs Throughput

Figure 3 shows that the test generated 270 hits per minute.

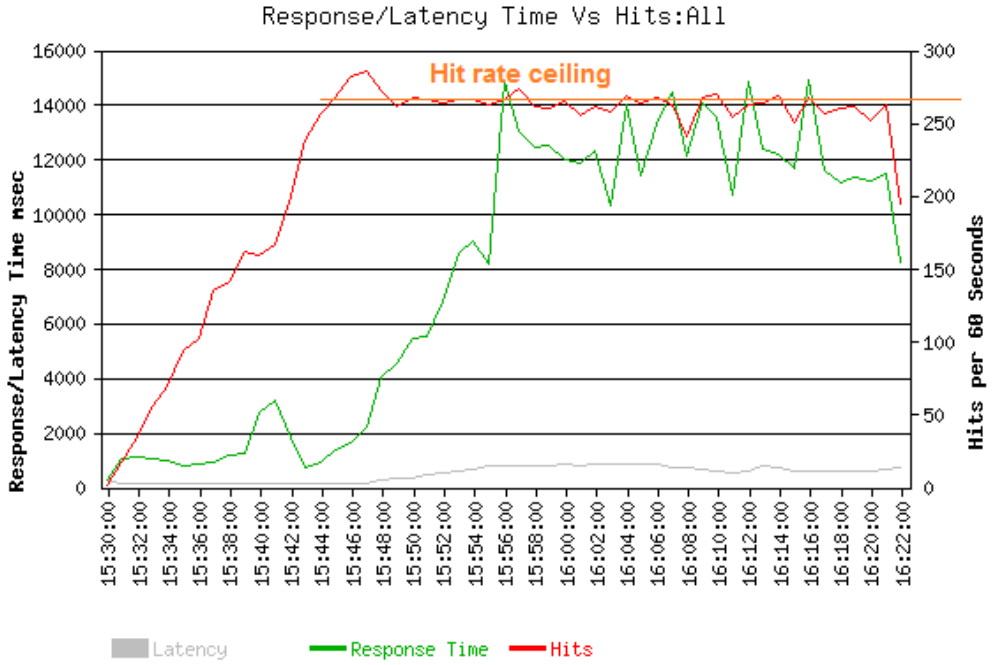

Figure 3 AWS instance 4 MB file– Response Time vs Hits

Although bandwidth is the obvious suspect to create the bottleneck, it is important to verify that it is actually the cause. To confirm that the bottleneck found was actually due to a bandwidth limitation we ran the same test with a smaller file. After running the two tests we can compare the maximum bandwidth and hit rate to determine which is responsible for the bottleneck.

For the next test we used 3 Megabyte instead of 4 Megabyte file.

The bandwidth bottleneck was still eminent. In this case again, it appears that the bandwidth provided by AWS was 530 Megabyte per minute.

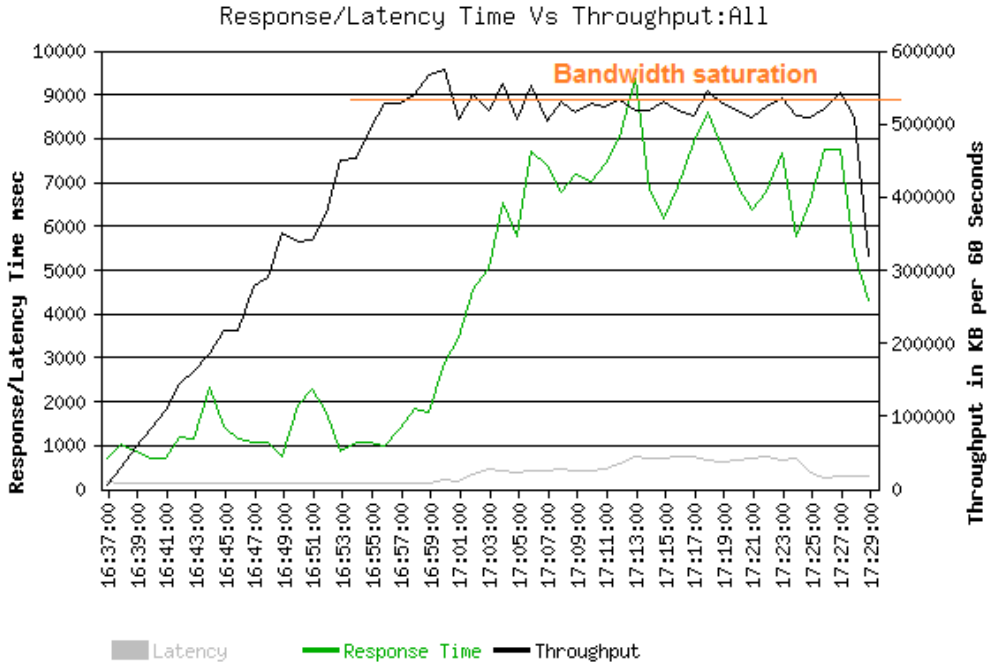

Figure 4 AWS instance 3 MB file– Response Time vs Throughput

Only this time when checking the hit rate, we see a different figure.

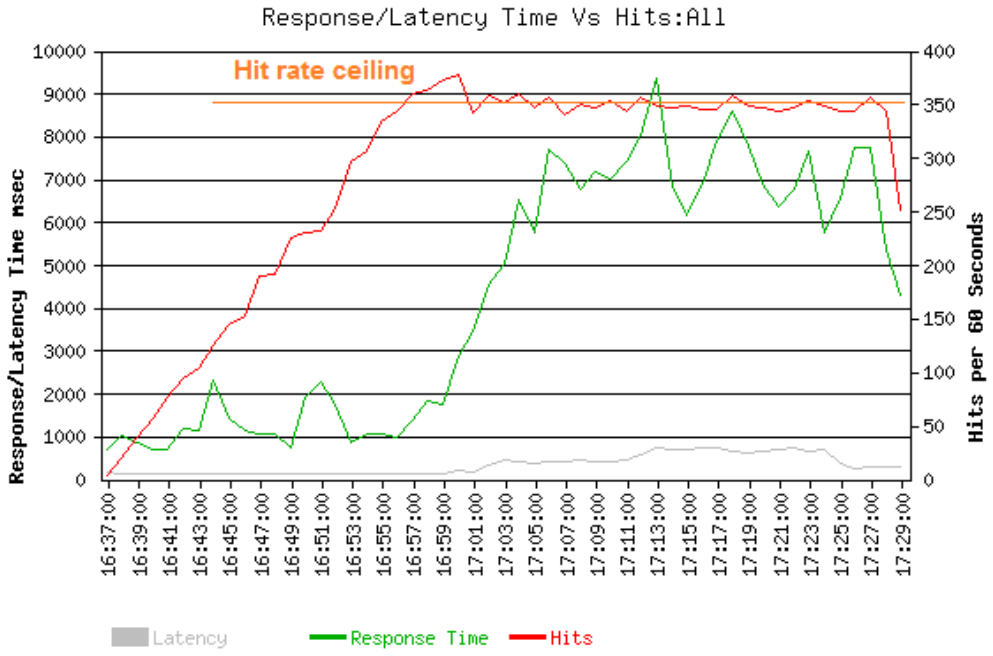

Figure 5 AWS instance 3 MB file– Response Time vs Hits

Looking at the hit rate graph from the last test, we can see that the hit rate reached a capacity of 350 hits per minute as opposed to the 270 hits per minute figure that was reached in the previous

test. As the maximum bandwidth was the same for the two tests and the hit rate varied, it is clear that the cause for the bottleneck is in fact bandwidth limitation.

Now, an alternative hypotheses is that this bandwidth limitation is not in the system under test, but for example in the test driver. However, this appears unlikely as the test driver is capable of generating a load of over 180 Mbit/sec (see the report on http://performancexpert.com/blog/how-analyze-results-large-scale-load-test).

Rackspace Results

The resource hosted at Rackspace provided a lower performance level in terms of bandwidth compared to the one hosted at AWS . It could only sustain about 20 concurrent users before the load presented its effects, as is shown in the next graph.

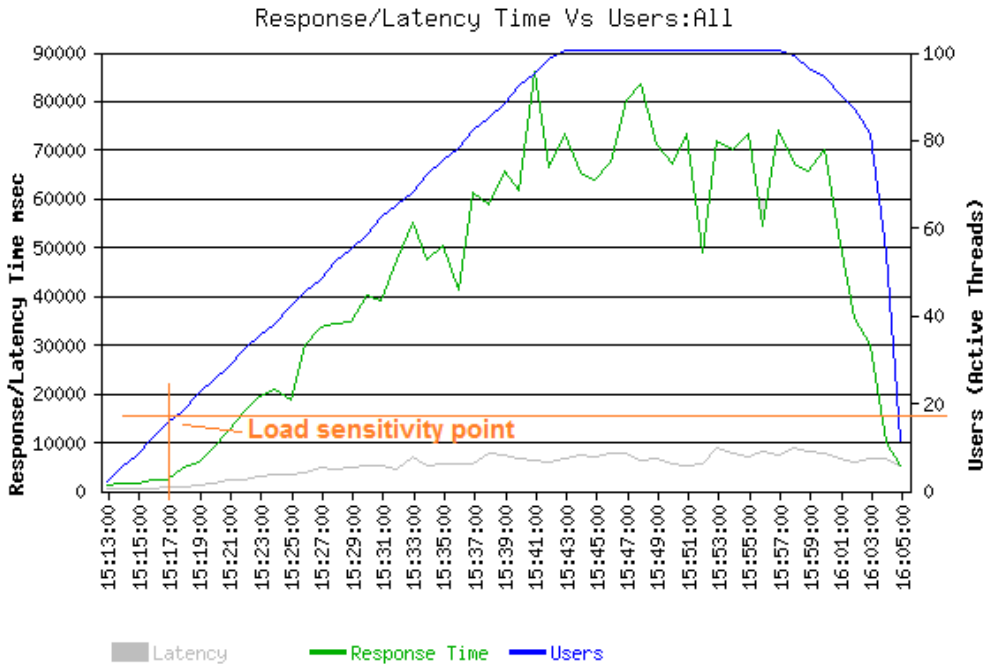

Figure 6 Rackspace instance 4 MB file– Response Time vs Simultaneous Users

The reason, again, was quite obvious.

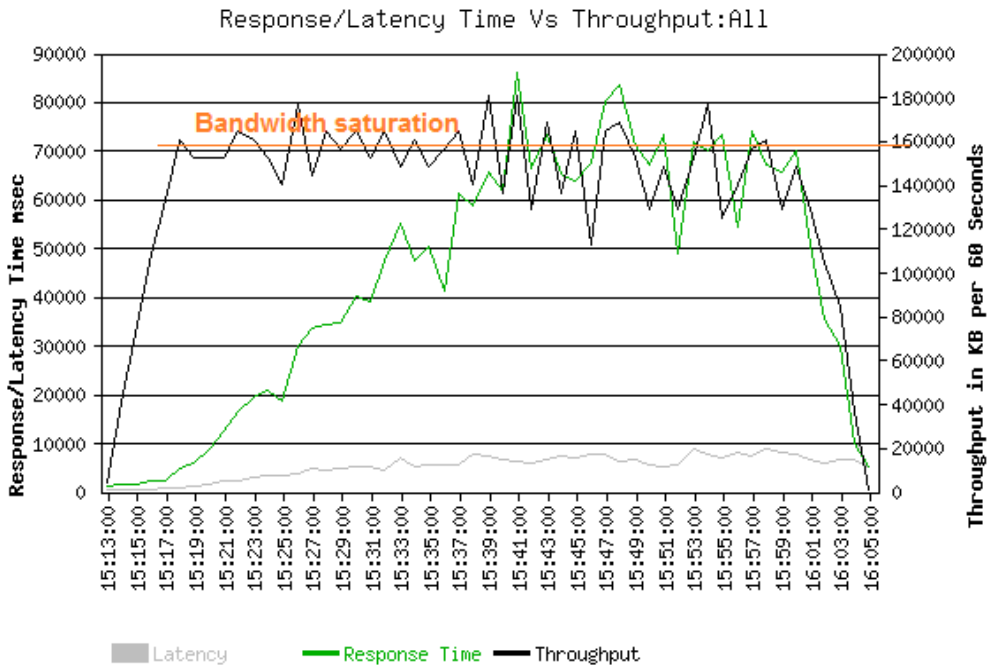

Figure 7 Rackspace instance 4 MB file– Response Time vs Throughput

In this case, it appears that the bandwidth provided by Rackspace was 160 Megabyte per minute, which is 21.3 Mbps. The last test generated close to 80 hits per minute.

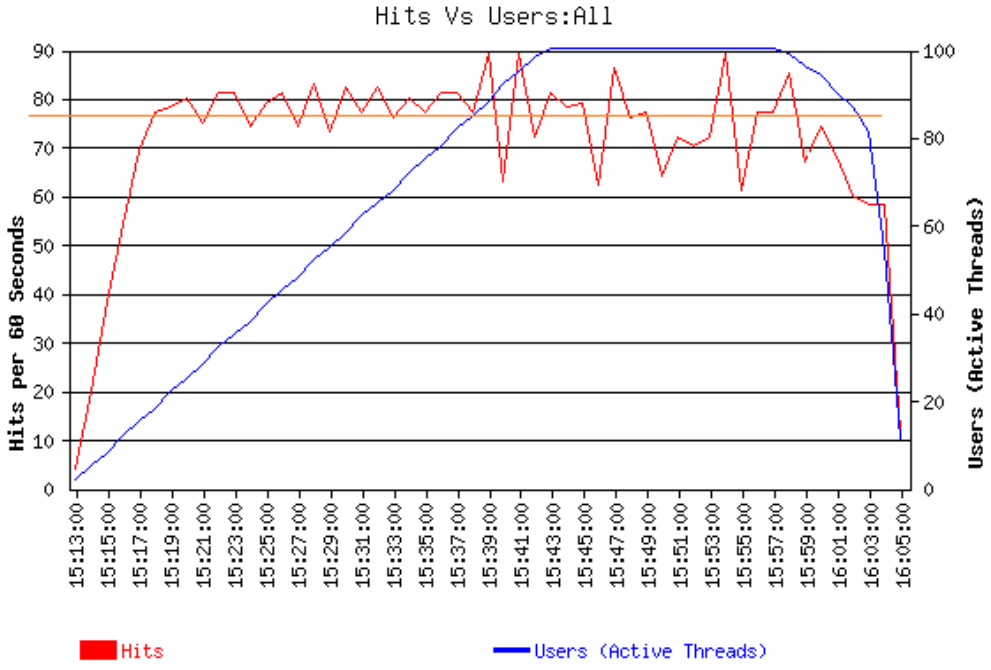

Figure 8 Rackspace instance 4 MB file– Hits vs Simultaneous Users

To verify that the cause for the bottleneck is bandwidth limitation, again, we ran the same test only with a smaller file.

After running the same test with a smaller file, 3 Megabyte instead of 4 Megabyte, we saw that the bandwidth bottleneck was still eminent. It appears that the bandwidth provided to this instance by Rackspace was close to 160 Megabyte per minute.

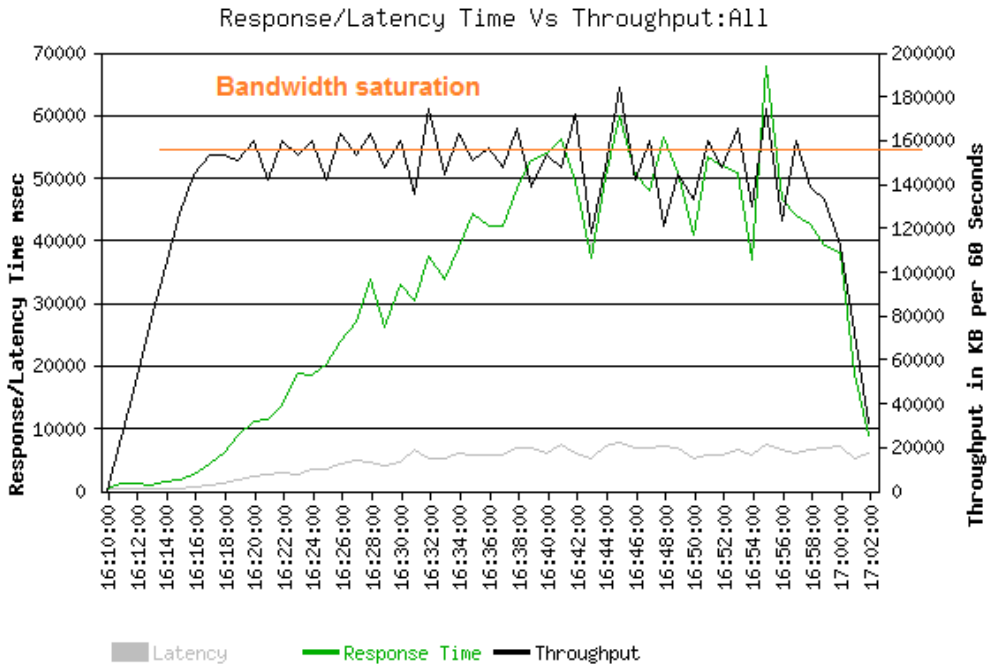

Figure 9 Rackspace instance 3 MB file– Response Time vs Throughput

When checking the hit rate, we see that maximum hit rate is just over 100 hits per minute as opposed to 80 hits per minute in the previous test.

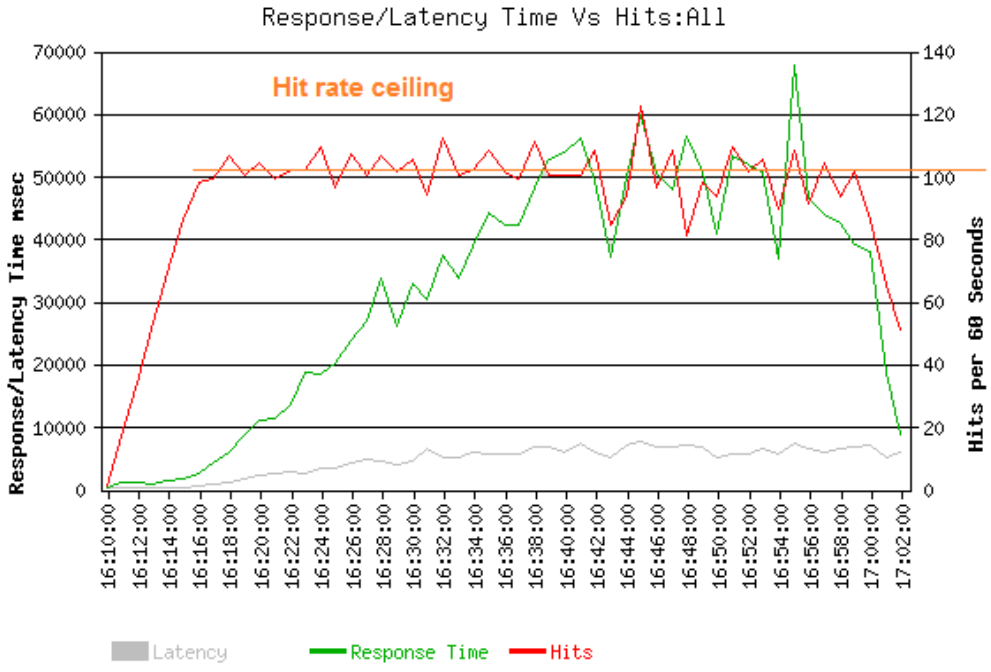

Figure 10 Rackspace instance 3 MB file– Response Time vs Hits

Here again, the last test confirms that the bottleneck results from a bandwidth limitation.

More details of the actual test results can be found through the following links.

Host	File size	URL of test results
AWS	4 Megabyte	http://performancexpert.com/report/8989
AWS	3 Megabyte	http://performancexpert.com/report/8994
Rackspace	4 Megabyte	http://performancexpert.com/report/8986
Rackspace	3 Megabyte	http://performancexpert.com/report/8992

Conclusions

Our measurements with identical loads show that

AWS provided bandwidth on a type Micro instance of 71 Megabit/second

Rackspace provided bandwidth on a type 512MB server of 21 Megabit/second

Your results may vary. Different server instance times may have different bandwidths available to them, and there may be more factors influencing bandwidth. The main contribution of this paper is to demonstrate a working approach to measure the bandwidth of cloud providers.