Understanding Network Communication Performance in Virtualized Cloud

Guohui Wang, T. S. Eugene Ng, Rice University, USA

1. Introduction

Cloud service allows enterprise class and individual users to acquire computing resources from large scale data centers of service providers. Users can rent machine instances with different capabilities as needed and pay at a certain per machine hour billing rate. Despite concerns about security and privacy, cloud service attracts much attention from both users and service providers. Recently, many companies, such as Amazon, Google and Microsoft, have launched their cloud service businesses. Users have also started to run a large variety of applications on cloud, such as content delivery, media hosting, web hosting and streaming processing.

Most cloud service providers use machine virtualization to provide flexible and cost-effective resource sharing among users. For example, both Amazon EC2 [1] and GoGrid [5] use Xen virtualization [3] to support multiple virtual machine instances on a single physical server. Virtual machine instances normally share physical processors and I/O interfaces with other instances. It is expected that virtualization can impact the computation and communication performance of cloud services. However, very few studies have been performed to understand the network performance of these large scale virtualized environments.

We study the network communication performance in virtualized cloud environments. The goal is to understand the impact of virtualization on network performance in cloud and its implications to cloud applications. We perform a large scale measurement study on the network performance of Amazon EC2 cloud. We measure the processor sharing, TCP/UDP throughput and end-to-end delays among Amazon EC2 virtual machines and observe that even though the data center network is lightly utilized, virtualization can still cause significant throughput instability and abnormal delay variations. We discuss the implications of unstable network on various cloud applications, especially on multimedia applications that requires stable and predictable network.

2. Network performance of Amazon EC2 cloud

Our measurement is based on small instances and high CPU medium instances on Amazon EC2 cloud. We perform a spatial experiment to study the network performance of 750 small instance pairs and 150 medium instance pairs at different network locations covering 177 subnets in Amazon EC2 us-east clouds. We also perform a temporal experiment to measure 6 small instance pairs and 3 medium instance pairs continuously over one week. In this section, we highlight some of our findings from the measurement study. Please refer to our full paper [6] for more details.

Processor sharing: we use a simple CPUTest program to test the processor sharing property of EC2 instances. This program consists of a loop that runs for 1 million times. In each iteration, the program simply gets the current time by calling gettimeofday() and saves the timestamp into a pre-allocated array in memory. When the loop finishes, the program dumps all the saved timestamps to the disk. Normally, if the program is executed continuously, all loop iterations should take a similar amount of time. However, virtual machine scheduling can cause some iterations to take much longer than the others. If the instance is scheduled off from the physical processor, we would observe a gap in the timestamp trace.

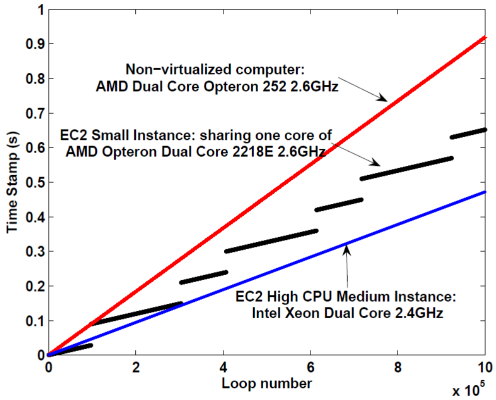

Figure 1. CPUTest timestamp trace

Figure 1 shows the CPUTest timestamp trace on different environments. When the CPUTest program is run on a non-virtualized machine or a medium instance, the timestamp traces produced indicate the CPUTest program achieves a steady execution rate with no significant interruption. However, the timestamp trace of the small instance shows very obvious scheduling effects.

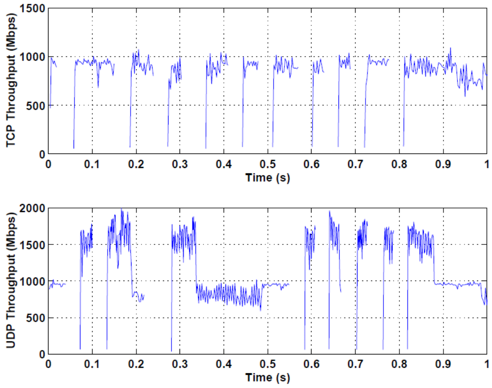

Figure 2. Fine grain TCP/UDP throughput trace for small instances

When the CPUTest program is run on a small instance, periodically there is a big timestamp gap between two adjacent loop iterations. The gaps are on the order of tens of milliseconds. We believe the on/off execution pattern is caused by virtual machine scheduling. The large timestamp gaps represent the periods when the instance running the CPUTest program is scheduled off from the physical processor.

From the timestamp trace of CPUTest program, we can estimate the CPU share of our virtual machine instances by compute the fraction of on period. Our statistics show that small instances always get 40-50% of CPU share, while medium instances mostly get the full CPU share. This means Amazon EC2 performs very restricted control on the processor sharing of small instances.

TCP/UDP throughput: Figure 2 demonstrates the fine-grain TCP and UDP throughput of a typical small instance pair. We consistently observe the same transmission pattern on all the small instances. To make the results clearly visible, we only pick one small instance pair and plot the throughput in a 1 second period. We observe drastically unstable TCP throughput, switching between full link rate at near 1 Gb/s and close to 0 Gb/s. The quiet periods last for tens of milliseconds. Considering the processor sharing behavior observed in our CPUTest experiments, we believe that the quiet periods are caused by the processor sharing among small instances. During these quiet periods, either the TCP sender instance or the receiver instance is scheduled off from the physical processor, therefore no packet can be sent out from the sender.

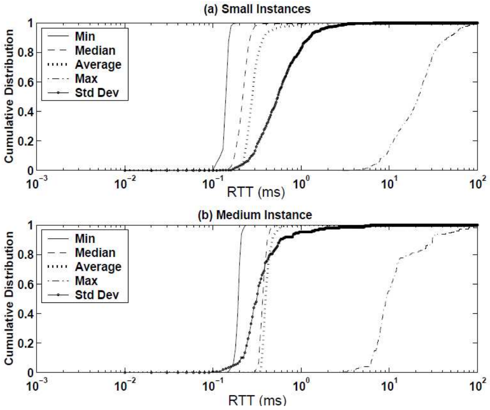

Figure 3. The distribution of delay statistical metrics

We observe a similar unstable UDP throughput on small instances. The difference between UDP and TCP transfers is that, in many cases, after a low throughput period, there is a period where the receiver receives UDP traffic at a high burst rate (even higher than the network's full link rate). We believe the reason is, during the low UDP throughput periods, the receiver is scheduled off from the processor, but the sender instance is scheduled on. All the UDP traffic sent to the receiver will be buffered in the Xen driver domain. When the receiver is scheduled on later, all the buffered data will be copied from driver domain memory to the receiver's memory. Since the data is copied from memory to memory, the receiver can get them at a much higher rate than the full link rate.

End-to-end delay: we measure the packet round trip delay (RTT) of 750 small instance pairs and 150 medium instance pairs using 5000 ping probes. For each instance pair, we compute the minimum, median, average, maximum RTTs and the RTT standard deviation from the probes. Figure 3 shows the cumulative distribution of these RTT statistical metrics for small and medium instances (note that the x-axis is in log scale).

From this graph, we can see that the delays among these instances are not stable. The minimum delays are smaller than 0.2 ms for most of the small instance pairs. However, on 55% of the small instance pairs, the maximum RTTs are higher than 20 ms. The standard deviation of RTTs is an order of magnitude larger than the minimum delay and the maximum RTTs are 100 times larger than the propagation delays. The delays of medium instances are more stable than the small instances.

But we still observe that, for 20% medium instance pairs, the maximum RTTs are larger than 10ms. Considering the Amazon EC2 cloud is based on a large cluster of computers in a data center, these large delay variations are abnormal.

3. Research issues

The measurement results from our study identify a few research questions that need to be addressed in cloud. For cloud service providers, the fundamental issue is how to balance the resource sharing and performance isolation in cloud. System designers need to improve the virtualization infrastructure for more stable and predictable network performance in cloud.

For cloud users, the unstable network can obviously degrade the performance of many data intensive applications. More importantly, many multimedia applications such as video processing and media hosting require stable and predictable network performance. The unstable throughput and abnormal delay variations raise significant challenges to run multimedia applications in cloud. The question is how we can adjust these applications when migrate them to cloud. Users may need to synchronize the sender and receivers to reduce the impact of virtual machine scheduling on unstable throughput.

Virtualization also makes it hard to infer the network congestion and bandwidth properties from end-to-end measurement. The abnormal variations in network performance measurements could also be detrimental to adaptive applications and protocols (e.g. TCP vegas [4], channel-adaptive video streaming [2]) that conduct network performance measurements for self-tuning. Given the observations from our measurement study, many adaptive applications need to be reconsidered to achieve optimal performance in virtualized cloud environments. Other than the network inference, diagnosing erroneous network behaviors from end-to-end measurement is also a challenging problem in virtualized cloud.

References

[1] Amazon EC2, http://aws.amazon.com/ec2/.
[2] B. Girod, M. Kalman, N. Liang, R. Zhang, Advances in channel-adaptive video streaming, in proceedings of ICIP'02, Dec, 2002.
[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, In *Proceedings of SOSP'03*, Oct. 2003.
[4] L. S. Brakmo and L. Peterson, TCP Vegas: End-to-end congestion avoidance on a global internet, *IEEE Journal of Selected Areas in Communication*, vol. 13, no. 8, Oct. 1995.
[5] Gogrid,. http://www.gogrid.com/.
[6] G. Wang, T. S. Eugene Ng, "The impact of virtualization on network performance of Amazon EC2 data center", in proceedings of IEEE INFOCOM'10, Mar. 2010

Guohui Wang is a final year Ph.D. student of Computer Science at Rice University. He got a MS from Chinese Academy of Sciences and a BS from University of Science and Technology of China. Guohui's research interests are on data center networking and cloud computing.

.

T. S. Eugene Ng is an Associate Professor of Computer Science at Rice University. He is a recipient of a NSF CAREER Award (2005) and an Alfred P. Sloan Fellowship (2009). He received a Ph.D. in Computer Science from Carnegie Mellon University in 2003. His research interest lies in developing new network models, network architectures, and holistic networked systems that enable a robust and manageable network infrastructure.