ACCELERATOR SIMULATION - BEYOND HIGH PERFORMANCE COMPUTING [*]

K. Song [†], K. Muriki, S. James, G. Jung, B. Li, Y. Qin, C. Sun, H. Nishimura
Lawrence Berkeley National Laboratory, One Cyclotron Road, Berkeley, CA 94720, USA

Abstract

Accelerator modeling and simulation studies heavily rely on High Performance Computing (HPC). Public Cloud computing has opened a new service horizon for HPC by offering an on-demand Virtual Private Cloud (VPC). Previously, we investigated using Amazon HPC public Cloud for lattice optimization applications and evaluated its performance. In this research, we use the Amazon VPC technology to study the feasibilities of extending local HPC resources into the Amazon Elastic Compute Cloud (EC2), and to provide a seamless, hybrid, and secure environment when the demand for computing capacity spikes.

INTRODUCTION

Accelerator modeling and simulation studies heavily rely on High Performance Computing (HPC). Public Cloud computing has opened a new service horizon for HPC by offering an on-demand Virtual Private Cloud (VPC). Previously, we investigated using Amazon HPC public Cloud for lattice optimization applications and evaluated its performance [1]. In this study, we work on extending a local computational cluster ALSACC into the Amazon Elastic Compute Cloud (EC2 [2]) by applying the Amazon Virtual Private Cloud (VPC) technology. ALSACC is a 28-node, 336-core Infiniband cluster used by the Accelerator Physics Group of the Advanced Light Source (ALS) at the Lawrence Berkeley National Laboratory. The CPU time of the cluster is mainly dedicated to the storage ring lattice optimization for the ALS future upgrades [3] and the injector optimization of the Next Generation Light Source (NGLS) [4]. The primary optimization algorithm is the Multi-objective and Multi-variable Genetic Algorithm (MOGA) [5], which involves evaluating the lattice properties thousands to millions times based on the optimization targets. The performance of the optimization highly depends on the computational power, especially to the injector optimization for the NGLS project. Therefore, extending ALSACC computing resource becomes essential for the further development of the NGLS in the next few years.

Amazon's VPC solution allows us to create an isolated segment of the Amazon Web Services (AWS). By using Amazon's Virtual Private Network (VPN) gateway one can connect this isolated AWS segment to the local private network securely. This allows us to extend local resources, such as compute nodes, storage targets, *etc.*, in the local private network into the public cloud. It also allows AWS resources launched inside of the isolated VPC segment, such as compute nodes, to become resources in the same local private network, therefore provides the possibility to extend services both ways. By using the Amazon VPC, one can expand local resources within a reasonable amount time based on demand. In this study we demonstrate extending a local computational cluster into the public cloud and compare the VPN performance between two different implementations.

ARCHITECTURE

To extend the local network into the Amazon EC2, two VPN endpoints are created locally and within the Amazon Cloud. A site-to-site VPN tunnel based on the Internet Protocol Security (IPSec) technology is created between these two endpoints, which allows communication to travel through this secure tunnel. Thus we can mount local storage targets remotely via NFS, and extend compute resources by adding remote nodes to the local scheduler, without affecting local user experience. Figure 1 illustrates the architecture of this technique. It is also worth of mentioning that creating the VPN tunnel and adding remote nodes could all happen behind the scene and be automated via on-demand request by extending the scheduler capabilities. Based on whether a physical appliance, such as the CISCO Integrated Services Router (ISR) or Adaptive Security Appliance (ASA), or a software solution, such as Openswan [6], is used or not on the endpoints, we can divide the implementations into four different categories: hardware-hardware, hardware-software, software-hardware, and software-software. In this study, we explored hardware-software (local-remote), and hardware-hardware (local-remote) solutions.

IMPLEMENTATIONS

Hardware-Software

In this implementation, CISCO 5520 ASA is configured as the local VPN gateway, while a compute instance running Openswan is configured as the remote VPN gateway. Based on the instructions provided by AWS, it is required to create a public subnet inside the VPC so that the Openswan instance is able to access the local VPN gateway. It is also required to create a virtual Internet gateway and route all the public traffic to go through this gateway. The next step is to allocate and associate an Elastic IP address to this instance thus one can access the Openswan gateway from the Internet. A private subnet is then created

[*] Work supported by U.S. DoE Contract No. DE-AC02-05CH11231

[†] KSong@lbl.gov

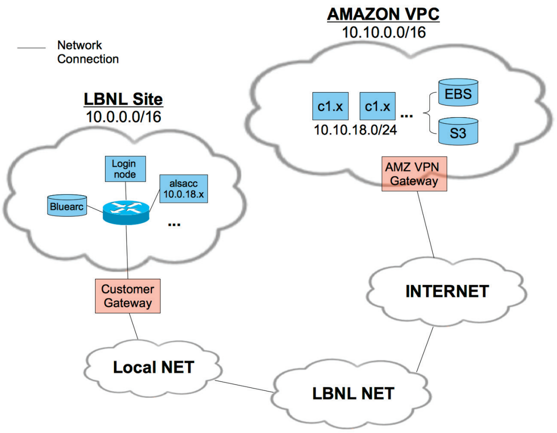

Figure 1: Architecture diagram for extending local resources into the Amazon EC2.

and is used to launch all the compute instances. The routing table for this subnet is updated to forward all the network traffic to the Openswan gateway. By using the Elastic IP address of the remote gateway, the local ASA device can be configured to create the VPN tunnel. Scripts based on our previous study [1, 7] were developed to automate this process. Once all the components are launched, compute instances in the VPC will be accessible to the local network. They can also mount local file systems similar to the ones demonstrated in the hardware-hardware implementation. It is noteworthy that the network address translation (NAT) technology is used at several locations in this implementation to make sure communication traffics are properly routed through various virtual gateways. This prevents the scheduler from managing compute instances properly because it requires a direct access to the clients on the compute instances instead of transversing through NAT. Hence in this implementation it is not possible to create a VPC that is completely identical to the local cluster.

Hardware-Hardware

Since the CISCO 5520 ASA is not supported by Amazon VPC service, a CISCO 1941 ISR is used as the customer gateway, and the hardware gateway provided by Amazon VPC service is used on the other side in this implementation. After the VPN tunnel is created, several EC2 compute instances are launched to create the VPC. Local file systems are mounted to these newly created compute instances and scheduler is updated to include these instances. At this stage, compute instances created inside of VPC are considered fully provisioned and are ready to schedule jobs similar to local compute nodes.

Amazon Machine Image (AMI)

One important task in building the VPC solution is to prepare the proper AMI to provision the compute instances. On the reference local HPC cluster, a similar technique called Virtual Node File System (VNFS) is used to provision compute nodes. It is natural to adopt the local VNFS and convert it to the AMI used on the EC2 instances so that a remote compute instance looks identical to a local compute node. To achieve this goal, some configuration files there are only used locally were firstly removed, and security challenges were updated to make sure that losing the AMI would not cause a severe security breach. A suitable kernel with kernel modules and ramdisk image provided by Amazon, to support the Paravirtual Machines (PVM) was integrated into the AMI. After the integration, an EC2 instance provisioned with this AMI looks similar to a local compute node.

PERFORMANCE

Performance evaluation appears to be a big challenge as the Amazon EC2 instances launched in this study are all standard instances (m1.large), which are different from the cluster compute instances (cc1.4xlarge) used in our previous study [1]. Due to this difference, comparing the performance of compute instances to local compute nodes makes it less interesting. Thus, in this research we focus on comparing the performance of the VPN tunnel which bridges the local cluster and the public cloud together. The performance of the secure tunnel is essential to a variety of services, such as storage, scheduler, DHCP, as well as external services such as egress accessibility, as all traffics are routed back to the primary gateway which is located locally.

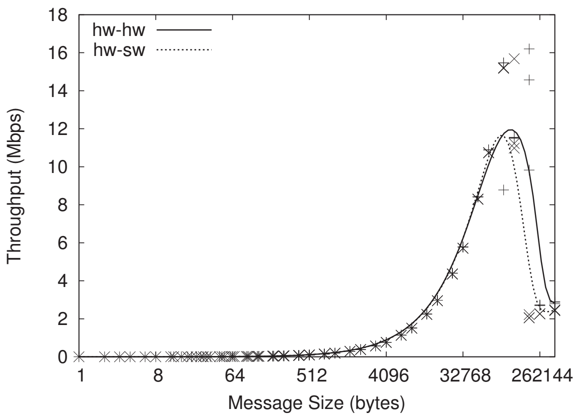

Figure 2: Throughput of the VPN tunnel in hardware-hardware and hardware-software solutions.

Figure 2 and 3 demonstrate the throughput bandwidth and latency measured under two aforementioned solutions. It is clear that the throughput bandwidth is higher while the latency is lower from the hardware-hardware solution than from the hardware-software solution at large message sizes. One explanation to this is that the traffic is offloaded to the hardware appliance instead of depending on the compute instance itself in the hardware-hardware solution, hence some performance improvement is observed

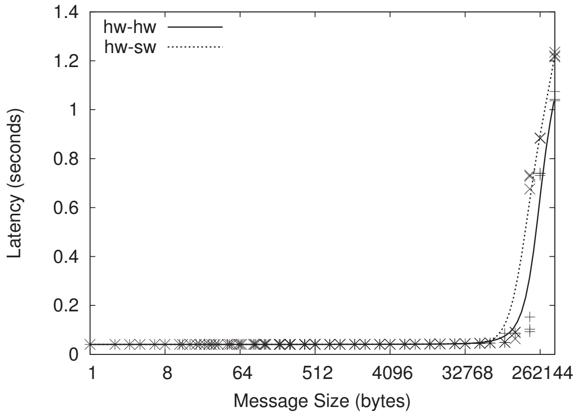

Figure 3: Latency of the VPN tunnel in hardware-hardware and hardware-software solutions.

since it does not consume system resources of the compute instance to pass traffic through the tunnel. In measurements, large variations to the bandwidth at very similar messages sizes were observed, *e.g.*, throughput at message sizes around 192 KB. The exact reason for such is not clear, however, it is possible that the bandwidth is shared among many Amazon AWS users instead of dedicated to our platform. The bandwidth drops significantly to about 2.8 Mbps after 256 KB of message sizes also further confirms this idea, as the high throughputs around 15 Mbps can be explained as spikes of variations, thus can be ignored for realistic and consistent runs. Larger than 256 KB of message sizes were also used to measure the consistent throughput and it matches the performance numbers showed in Fig. 2 and 3. The ping-pong latency is stable at about 40 ms, which appears to be a challenge for latency sensitive activities, such as interactive activities. Storage performance is also a big concern as all the storage traffic is routed back to the central storage targets in this study. Comparing to the 40 Gbps of bandwidth and 2 µs latency on the local cluster, it is clear that it is insufficient to run large parallel jobs across this VPN tunnel. One potential usage to this extension is similar to the research we demonstrated earlier [1] - to run separate jobs on the cloud itself. However, during the testing of running medium size parallel jobs across the compute instances in the VPC, network stability issues were observed so jobs were not able to finish cleanly. Hence, if there is no significant improvement in the upcoming VPC solution for the cluster compute instance, we could still suffer similar network performance through the VPN tunnel. One solution to this is to provide a dedicate bandwidth by using some of the advanced bandwidth sharing technologies, such as quality of service (QoS) to guarantee the delivery of traffic.

SUMMARY

In this preliminary research, we studied two ways of creating a secure tunnel to integrate a remote virtual private cluster into the existing local HPC cluster. This study is meant to be a proof of concept research instead of a production implementation. Results from this research show that it is possible to integrate the Amazon EC2 instances into a local HPC cluster. However, building such a solution is not straightforward with the tools that Amazon provides. It is also limited by the technologies available, such as NAT that is demonstrated in this paper. The solution itself should also vary case by case as different organizations would tend to have different methods to manage their resources, in terms of provisioning the system, scheduling jobs across different resources, deploying storage targets, updating security policies, *etc*. We also measure the network performance through the VPN tunnel and it is still premature to assume it is capable of running parallel jobs across the virtual cluster. When the VPC solution for the cluster compute instance is available, and the network performance is greatly improved at a later time, we should revisit this solution to find possible applications for it.

ACKNOWLEDGMENTS

This research used resources provided by Laboratory Research Computing (LRC) [8], which is managed by the LBNL IT Division. The authors would like to thank D. Robin and J. Kekos from ALS for their support and funding; members of High Performance Computing Services group in the IT Division, G. Kurtzer, J. White, J. Scoggins, and M. Jennings for their technical assistance; and the LBNL Network Services group, especially K. Schafer, for their loaner devices and support.

REFERENCES

[1] C. Sun *et al.*, “HPC Cloud Applied to Lattice Optimization”, Proceedings of PAC 2011, New York, WEP151.

[2] Amazon Elastic Compute Cloud (Amazon EC2), `http://aws.amazon.com/ec2`.

[3] C. Sun *et al.*, “Small-Emittance and Low-Beta Lattice Designs and Optimizations”, Phys. Rev. ST Accel. Beams 15, 054001 (2012).

[4] C.F. Papadopoulos *et al.*, “Multiobjective Optimization for the Advanced Photoinjector Experiment (APEX)”, Proceedings of FEL2010, Malmö, Sweden, WEPB37.

[5] K. Deb, IEEE transactions on evolutionary computation, vol. 6, no. 2, 2002, pp.192-197.

[6] Openswan project, `https://www.openswan.org`.

[7] K. Jackson *et al.*, “Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud”, 2^{nd} IEEE International Conference on Cloud Computing Technology and Science, Indianapolis, 2010.

[8] Laboratory Research Computing (LRC), `http://lrc.lbl.gov`.