CloudTrace Demo: Tracing Cloud Network Delay

Giuseppe Di Lena, Frédéric Giroire, Thierry Turletti, Chidung Lac

CloudTrace Demo: Tracing Cloud Network Delay

Giuseppe Di Lena
Université Côte d'Azur, Orange Labs
Lannion, France
giuseppe.dilena@orange.com

Frédéric Giroire
Université Côte d'Azur, CNRS
Sophia Antipolis, France
frederic.giroire@cnrs.fr

Thierry Turletti
Inria, Université Côte d'Azur
Sophia Antipolis, France
thierry.turletti@inria.fr

Chidung Lac
Orange Labs
Lannion, France
chidung.lac@orange.com

***Abstract*—Many companies and organizations are moving their applications from on-premises data centers to the cloud. The cloud infrastructures can potentially provide an infinite amount of computation (e.g., Elastic Compute) and storage (e.g., Simple Service Storage). In addition, all cloud providers propose different offers: IaaS, PaaS, and SaaS. This demo focuses on the IaaS services, presenting a simple tool to measure the network delay in a virtual infrastructure built entirely in the cloud. These measurements are useful for organizations that are moving current applications to, or creating new applications in, the cloud, but have requirements on the maximum, or average, network delay that these applications can tolerate. We present CloudTrace, a simple CLI tool that creates regional and multiregional experiments to measure delay, using Amazon AWS.**

***Index Terms*—Cloud, Network**

I. Introduction

Cloud computing provides convenient on-demand access to a potentially unlimited pool of computing resources. In recent years, cloud computing resources have become cheaper and more powerful, thanks to the growing interest of companies around the world. Cloud computing brings several advantages: a company has little or no capital expenditure (CAPEX) when launching a new product, instead of having to invest in data centers and servers before knowing how much these resources are going to be used. It is flexible and on-demand. This means that the users can consume computing power and only pay for the amount of resources they actually consumed.

On a high level, cloud providers offer three service models, Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). We focus on IaaS, where users can manage the virtual instances and networks created in the cloud infrastructure, while the cloud provider is responsible for managing the physical infrastructure for providing a virtualization layer to isolate each user's environment.

We provide a brief overview of the main component of Amazon Web Services (AWS) that is responsible for this cloud model which is Amazon Elastic Cloud Computing (EC2). We describe all the virtual devices involved in the creation of a virtual infrastructure, and we apply the best practice strategies to deploy our experiments in a regional and multiregional environment. All major cloud providers offer resources for each instance flavor (e.g., t3.medium, m5.large, etc.), describing how many vCores and how much RAM are assigned. Regarding networking, AWS provides the maximum throughput that a flavor is able to generate (usually from 500 Mbps to 100 Gbps). AWS does not provide the maximum delay inside the infrastructure. It only mentions that all the regions and the availability zones are connected with a high performance optical fiber network.

978-1-6654-0522-5/21/$31.00 ©2021 IEEE

The goal of this demo is to present a simple Command Line Interface (CLI) tool that can measure the network delay between two, or more, different networks in AWS using different deployment strategies.

II. Related work

Since AWS launches EC2 service [2], multiple approaches have been proposed to measure computing or network performances in virtualized environments and cloud infrastructures. Most of them focus on the impact of a virtualized environment with different hypervisors [10], virtualization using Virtual Machines and containers [9], or the network performances variability in virtualized instances [5]. One of the most convincing measurement tools for the cloud environments is Cloudbench (CBTOOL) from IBM [7]. This tool automates experiments for a large range of workloads (e.g., Hadoop [6]) in cloud environments. In [8], the authors performed large-scale traceroute measurements over the global AWS network infrastructure. The traceroutes were performed between 15 regions, and the goal was to study the complexity of the AWS infrastructure. The work in [4] covers more generally the major cloud providers and the main Tier-1 ISPs. It shows how Internet traffic is more and more generated and shared with private interconnection between the cloud providers, thus bypassing the Tier-1 ISPs.

III. AWS Background

The AWS global infrastructure is composed of 2 different entities: **Region** and **Availability Zone (AZ)**.

The Availability Zone is a set of one or more data centers connected by redundant high-speed networks in order to guarantee high availability to the users. The region represents a set of multiple AZs clustered in a specific area and located around the world, within 100 km of each other. AZs in a same region are interconnected with high-bandwidth, low-latency, high-throughput networks, over fully redundant and dedicated metro fiber. To guarantee privacy, all traffic between AZs is encrypted. If an application is partitioned across AZs, companies are better isolated and protected from issues such as power outages or natural disasters [1]. AWS provides hundreds of different services. We describe briefly the main services and tools provided for IaaS. The first layer of an IaaS in AWS is the **Virtual Private Cloud (VPC)** that creates a

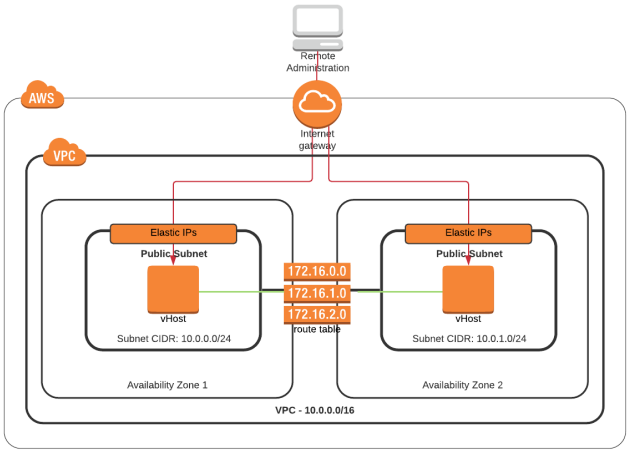

Fig. 1. Architecture in regional setup

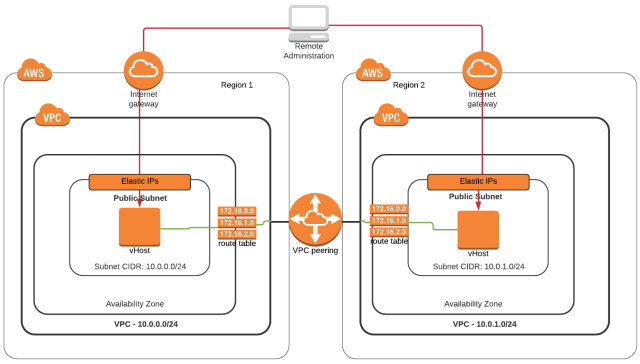

Fig. 2. Architecture in multiregional setup with VPC peering

virtual network defined by the user. All virtual instances and virtual devices (i.e., virtual gateways or route tables) should be created within a VPC. The VPC is composed of one or more **subnets**. The user managing the subnets can create as many subnets as needed in a single VPC till the address space in the VPC becomes empty. To control network flows in the network subnet, the user can create **route tables** associated with them. If the instances are in the same VPC, it is possible to connect them directly using the private IP if they are in the same subnet. If they are in a different subnet, the route table should be configured before the connection (Fig. 1). By default, the VPC does not provide any way to connect the instances to the external network. The **Virtual Internet Gateway** is a highly scalable virtual device managed entirely by AWS that allows traffic to go from the VPC to the outside world and vice-versa. Once created, the route table should be updated to redirect traffic to the subnets. Two other main services are **Network Access Control List (NACL)** and **Security Group (SG)**. NACL is a stateless firewall associated with an entire subnet, while SG is a stateful firewall associated with a virtual interface. There are three types of address in AWS: private, public, and **Elastic IP**. Elastic IP is a public address that can be conveniently attached to, and detached from, a virtual interface at any time. Once created and attached, the Elastic IP is not released even if the user deletes the instance or the interface. The address should be explicitly released by the user when it is not needed anymore. AWS proposes the **VPC peering** service [3] which makes it possible to connect two or multiple VPCs. Instances in the peering can connect between them via private IP, as if they are in the same network. Amazon ensures that the traffic going via the peering never leaves the AWS network, increasing security and performances. The last service is **Elastic Compute (EC2)**, that provides secure and resizable compute capacity in the cloud, and proposes a large number of instance types [2]. For our experiments, we use different architectures (Figs. 1 and 2). Fig. 1 describes a regional infrastructure composed of two subnets created in two different AZs within a single VPC, i.e., in a single region. As explained above, the Internet Gateway is responsible to allow connectivity with the external Internet. An instance is created on each subnet, and the flow table is stored in a single route table, serving the two subnets. The instances can connect to each other via the public IPs (an elastic IP is attached to each interface of the instance), or via the private IP since they are on the same VPC. In Fig. 2, the infrastructure is shared between two different regions. The architecture is similar, and the instances can still use the public IP to connect to each other like in the regional environment, but they cannot use by default the private IPs. In this case, a VPC peering connection between the two VPCs needs to be created to allow the instances to use the private IP to connect them, as if they were on the same network. Our tool automatically generates the infrastructure in both cases, with two or more AZs in case of regional and two or more regions in case of multiregional experiments.

IV. Implementation

CloudTrace is open-source, and it is compatible with Linux and MAC OSX, while it is possible to install it on Windows via Docker. The tool is built using LiteSQL, Ansible, Paris-Traceroute, and the Boto3 API for the automatic deployment in Amazon AWS. The basic idea is to have a CLI that takes as input the regions and the type of experiment that the user wants to run, and creates an environment performing multiple traceroutes between the virtual instances. First of all, CloudTrace creates a unique ID for the experiment (a string composed of 8 alphanumeric random characters).

Regional Deployment. The user has to specify only one region. The tool first checks if there is at least one VPC slot available in the region, then creates the VPC with subnet *10.0.0.0/16*. Finally, it creates then a *10.0.X.0/24* subnet for each Availability Zone (AZ) available in the region. After setting up the subnets, it creates an Internet gateway and attaches it to the VPC. It then creates a route table and appends all the routes necessary to allow external connectivity to the future instances created within the subnet. To allow Internet connectivity, adding the route is not enough. The tool also adds the rules in the NACL. After configuring all the subnets

and the NACL and SG rules, the tool creates an instance in each subnet.

Multiregional Deployment. The setup of the infrastructure is similar to the regional one. First of all, the tool connects with each region specified by the user and checks if there is at least one VPC slot for each region. If it is the case, it creates a VPC, a subnet and an Internet gateway in each region in the default AZ, and configures them like in the regional experiment. The VPC and the virtual subnet have the same subnet *10.0.X.0/24*, because in case of VPC peering, the subnet cannot overlap.

If the user selects the option to use VPC peering, a peering connection between all the VPCs are created, and all the route tables are updated, in order to forward the traffic via the private IP between remote VPCs (Fig. 2). All the information needed is saved in the client database (i.e., public IPs, VPC ID, regions, instances ID, etc.). The tool parses all this information and creates for each experiment an Ansible **inventory** (a file that describes login info and addresses of the instances). For each experiment, Ansible runs a standard **playbook**, a YAML file that lists all the requirements the instances have to install and all the commands that each instance has to execute before running the experiment (e.g., update the kernel).

V. Experiments

We present two experiments, tracing the network performances during one month in Amazon AWS, we will present in the demo how easily is to setup multiple experiments like these. The tests are made on two AWS regions: Frankfurt (eu-central-1) and London (eu-west-2), and ran from 04/11/2020 till 04/12/2020, where we collected in total more than one millions of traceroutes.

Regional Experiment		Confidence Interval Delay [ms]	
Source	Destination	Public IPs	Private IPs
eu-central-1a	eu-central-1b	0.835 ± 0.697	1.027 ± 1.04
eu-central-1b	eu-central-1a	0.864 ± 0.839	1.147 ± 1.202
eu-central-1a	eu-central-1c	1.066 ± 0.7	1.301 ± 1.496
eu-central-1c	eu-central-1a	1.094 ± 0.599	1.188 ± 1.369
eu-central-1b	eu-central-1c	0.915 ± 0.928	1.122 ± 1.505
eu-central-1c	eu-central-1b	0.943 ± 0.961	1.482 ± 2.149

TABLE I
Frankfurt delay with confidence interval (99%) with public and private IPs

Public vs. Private. AWS makes it possible to create a VPC peering between two or more different regions. This is done to connect two instances deployed in separated regions with the private IP assigned by AWS. As we can see, the delay using the private IP is higher, and also less stable, than for the public IP. This behavior is also repeated in all the regional experiments we performed. For example, Table I shows the delay with the confidence intervals in the Frankfurt region. We think that these differences are due to the additional layer added inside the AWS network and the control that each packet has to pass in order to circulate inside the AWS network.

Multiregional Experiments. We are performing the experiment using the public IP and AZ in different regions. Multiregional experiments mean long distances since the traffic has to go from Frankfurt to London, i.e., higher delays are expected.

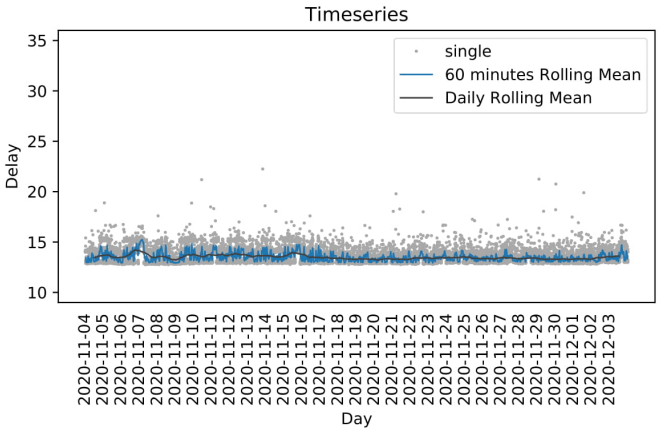

Fig. 3. Eu-central-1a to eu-west-2a trace (public IP)

Fig. 3 shows the delay between Frankfurt and London. The black line indicates the daily rolling mean (it includes the mean of the last 24 hours), while the blue line shows the hourly rolling mean (the last 60 minutes).

VI. Conclusions

AWS provides hundreds of services to their customers. We considered the main EC2 components (e.g., VPCs, subnets, Virtual Internet Gateway) that companies and organizations use to move their infrastructures to the cloud. We mainly focused on the networking performances of the global infrastructure. Companies that are moving delay-sensitive applications on AWS are interested in the latency of the network in a high availability (multiregional) configuration or on a regional configuration with multiple Availability Zones. We implemented CloudTrace, a simple tool to measure the networking performances and automatically plot statistics, together with a map of the analyzed network. CloudTrace is publicly accessible at https://github.com/Giuseppe1992/CloudTrace.

References

[1] Amazon. AWS Global Infrastructure documentation. https://aws.amazon.com/about-aws/global-infrastructure/regions_az/, 2020.

[2] Amazon. EC2 instance types. https://aws.amazon.com/ec2/instance-types/, 2020.

[3] Amazon. Vpc peering aws documentation. https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html, 2020.

[4] Todd Arnold, Jia He, et al. Cloud Provider Connectivity in the Flat Internet. In *ACM IMC*, pages 230–246, 2020.

[5] Karyna Gogunska, Chadi Barakat, and Guillaume Urvoy-Keller. Tuning optimal traffic measurement parameters in virtual networks with machine learning. In *IEEE CloudNet*, pages 1–3, 2019.

[6] Hadoop. Hadoop docs. https://hadoop.apache.org/docs/stable/, 2020.

[7] IBM. IBM Cloudbench Documentation. https://developer.ibm.com/depmodels/cloud/projects/cloudbench-cbtool/, 2020.

[8] Quentin Jacquemart, Alessandro Baldi Vitali, and Guillaume Urvoy-Keller. Measuring the Amazon Web Services (AWS) WAN Infrastructure. In *CoRes 2019*, Saint Laurent de la Cabrerisse, France, 2019.

[9] Z. Li, M. Kihl, Q. Lu, and J. A. Andersson. Performance overhead comparison between hypervisor and container based virtualization. In *IEEE AINA*, pages 955–962, 2017.

[10] PV Vardhan Reddy and Lakshmi Rajamani. Evaluation of different hypervisors performance in the private cloud with SIGAR framework. *IJACSA*, 5(2), 2014.