Lab: intro to Compute and Networking Services

AWS compute services

EC2

Allow you to launch on-demand virtual machines.

EC2 autoscaling

Allow you to dynamically scale EC2 capacity up or down according to conditions you defined. Scale up by launching more EC2 instances, and scale down by terminating EC2 instances that you don't need.

The condition you can pre-define such as CPU usage which is typically a good metric to use for autoscaling.

Amazon Lightsail

Easy way to launch virtual server to run an application. It will configure DNS management and storage for you

Elastic container service

Container management service for docker containers. They will be run in a EC2 instances.

AWS Lambda

Serverless service, allow you to run code without worrying about provisioning and managing the servers at all. You can just focus completely on writing the business logic and the code, just have to upload it and you're good. It runs on pay as you call rather than by the hourly. If the function isn't invoked at all then you won't be billed for it.

Webserver example

Vertical scaling: Say for example your current EC2 web server instance is being overwhelmed by requests. You will need to scale somehow. The old way 10/20 years ago to scale is by taking down your EC2 instance and then deploy a new one that is bigger and more powerful.

But this takes time, and your application is not running. What if the demand is only a peak and goes back down after. So vertical scaling doesn't handle that really well.

Horizontal scaling: To solve the problem that vertical scaling cannot handle we use horizontal scaling. We add more instance of the same EC2 instance to handle those demand, and as those demand decreases we terminate those instances.

Our application will still be up. However, the problem with horizontal scaling is that customers aren't going to know which EC2 instance is up and down when demand goes up and down. We will need a load balancer, and in AWS, Elastic Load Balancer will keep track of all the EC2 instances and distribute the incoming request to the available EC2 instances.

Auto scaling services: It will launch and terminate EC2 instances for you automatically according to the demand. It can also do health check, replacing unhealthy EC2 instances for you automatically.

Networking & content delivery

CloudFront

Securely delivery frequently accessed content, similar to a cache at a high transfer speed for your end-users. It also provides protection against DDos attack. It can reach many edge locations that are far from data centers.

Virtual private cloud

Rather than a traditional network where the network are done via hardware switchers and routers, there is another way of creating a computer network, which is via software network, or virtual network.

VPC is the AWS implementation of the virtual network, allows you to allocate a private section on the AWS cloud where you can launch your AWS instances into and interact with each other or when allowed interact with the public internet. Without any security group, **no traffics are allowed in to a VPC, but all traffics are allowed out from a VPC.**

Think of VPC as your own personal space that no one can enter without you explicitly allowing it.

Direct connect

A dedicated high speed network connection to AWS from on-premise data centers.

Elastic load balancing

Automatically distribute incoming traffics for your application across multiple EC2 instances and also in multiple availability zones. So if one availability zone goes down, it will automatically route to other availability zone, and because data are replicated and redundant in availability zone, it is okay if the traffic is routed to other availability zone.

Let you achieve high availability and fault tolerance by distributing traffic evenly among those instances.

Route 53

Highly scalable and available DNS server. Direct your domain name to say a backend web server. A DNS server basically.

API gateway

Easy for developer to create and deploy secure API at any scale. Handle tasks of accepting and processing up to hundreds and thousands of concurrent API calls.

Serverless service, no need to worry about provisioning the servers underneath.

CDN Example

You can leave static files such as large videos and images to CloudFront CDN to handle rather than passing it to a EC2 instance to handle. Let EC2 instances handle those dynamic content request that aren't static and require a server to compute and handle that request for the user.

You would set CloudFront as the entry point, let it serve static content and forward request for dynamic content to the load balancer which will distribute the request evenly among EC2 instances.

CloudFront CDN will generate a complicated domain name that is associated to that CloudFront instances, you are not going to give that to the end user right? They are not going to be remember it, and to solve that we can use Route 53 to map the domain name of your application to the CDN.

Revision #1
Created 2023-01-10 20:34:48 UTC by Tamarine
Updated 2023-02-06 18:24:02 UTC by Tamarine