Laboratorio 2: Terraform y LBaaS (Balanceadores de carga como servicio)

Benito Luongo Vegas, beluonve@alumni.uv.es

12 de marzo de 2023

Índice

1. Listado de capturas

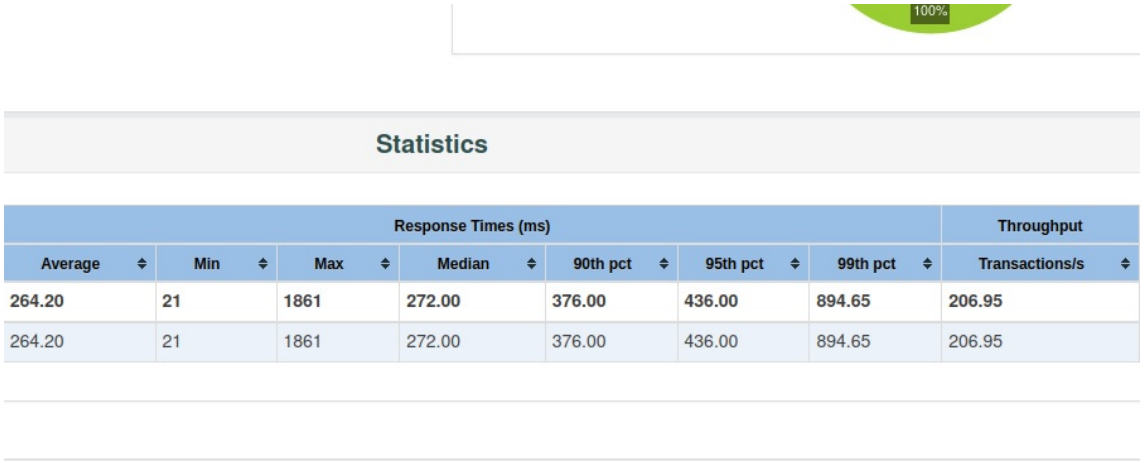

Figura 1: Una instancia y 100 usuarios

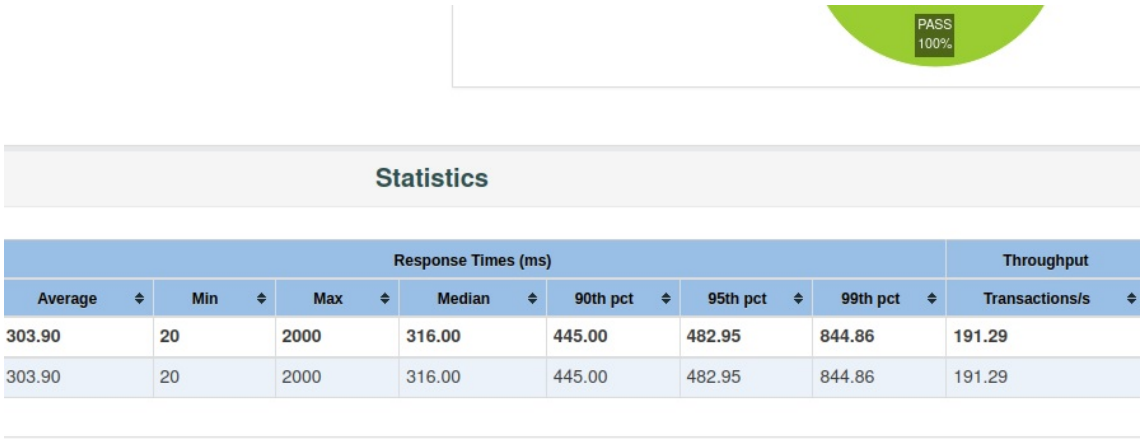

Figura 2: Dos instancias y 100 usuarios

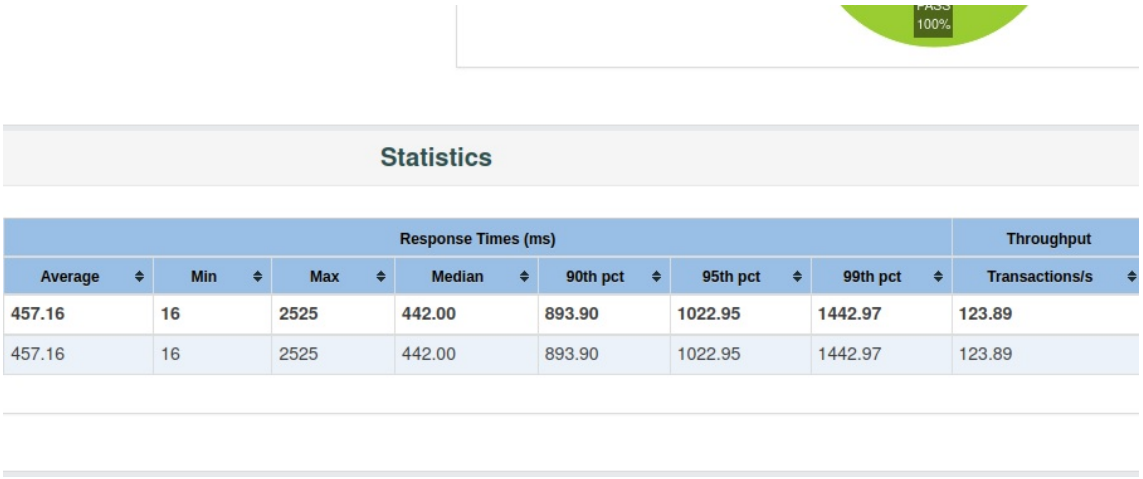

Figura 3: Tres instancias y 100 usuarios

2. Introducción

En este laboratorio se tendrá como objetivo el despliegue de una infraestructura como servicio utilizando la herramienta de automatización Terraform al igual que una integración de Balanceador de carga como servicio utilizando los comandos de OpenStack Neutron lbaas.

Adicionalmente, se comprobará el rendimiento, con el uso de JMeter, de tres diferentes servidores nginx con texto plano, residiendo en un pool dentro del balanceador de carga.

3. Instrucciones de Ejecución

Para realizar la creación de los elementos de la LBaaS (Balanceador de carga como servicio), primero se debe realizar una ejecución del contenedor con la imagen proporcionada:

Listing 1: Ejecutar contenedor a partir de imagen

```
docker run --name infra-as-code -it \
--add-host controller:147.156.84.206 \
-v /home/twcam/cloud/openstackrc:/tmp/tempdir \
twcammaster.uv.es/infra-as-code:latest /bin/bash
```

Al estar dentro, veremos que hay tres directorios con scripts, uno con los test plans y resultados/outputs de JMeter, otro con los scripts para la creación del balanceador de carga y finalmente el directorio con todos los ficheros relacionados a terraform para el despliegue de la infraestructuras

3.1. Terraform

Primero, para desplegar la infraestructura, tenemos que entrar en el directorio de Terraform y realizar un *source* al fichero de configuración de openstack. Luego debemos de realizar `terraform init` para cargar cualquier módulo nuevo que haya podido ser incluído. Para observar los cambios que se van a realizar en el proceso de automatización, ejecutamos `terraform plan -var-file cluster.tfvars`. Si estamos contentos con los resultados, simplemente se especifica `terraform apply -var-file culster.tfvars` para realizar toda la gestión de redes, servidores, ip's flotantes y demás.

3.2. Openstack-cli

Para la ejecución del balanceador de carga, se abre el directorio de *openstack-cli* en donde realizamos el *source* al fichero de configuración de openstack y ya podremos ejecutar `./crea-loadbalancer <nombre_de_projecto>` para desplegar todos los elementos de la infraestructura como es el balanceador de carga en sí, el listener, pool de miembros con sus respectivos monitores de estado. De la misma manera, para desmantelar los elementos del balanceador de carga, ejecutamos `./delete-loadbalancer <nombre_de_projecto>`

3.3. JMeter

Dentro del GUI de Jmeter escogemos la opción de tools del menú y seleccionamos import from cURL. Al generar el test, agregamos variables para el número de usuarios, de iteraciones

Listing 2: Ejecución del test plan de JMeter

```
./jmeter -n -t load-test.jmx -Jthread.users=100 \
-Jiterations=50 -Jduration=130 -l \
resultados/three_hundred.jtl \
-e -o outputs/three_members/hundred
```

También podemos especificar el ip flotante con `-Jfloating_ip=147.156.86.10`

Adicionalmente, se pueden generar resultados por medio de un fichero properties que especifica el número de usuarios e iteraciones. Sin embargo, este no resulta en un output legible, si no que genera un extenso fichero .jtl en el que luego podemos subdividir cada resultado en el threadgroup que corresponde

Listing 3: Ejecución del test plan de JMeter

```
./jmeter -n -t LoadBalancer_Test_Plan.jmx \
-q userThreads.properties -Jduration=130 -l \
resultados/three_hundred.jtl \
```

4. Rendimiento

Se puede observar en las figuras, que aunque la carga de peticiones se subdivida en varios servidores, no muestra ninguna diferencia notable, más bien, a medida que tenemos más instancias, menor son las transacciones por segundos y mayor es el tiempo de espera promedio, algo que es sumamente contraintuitivo

5. Conclusiones

En conclusión, el uso de herramientas como Terraform facilita no solo el despliegue, si no la eliminación de elementos en una infraestructura. La versatilidad que posee Terraform habilita a los usuarios escoger de entre los proveedores de cloud más usados y con acceso a comandos para instanciar recursos y modulos que no se diferencian mucho entre s.

Bibliografía

[1] HashiCorp, *Openstack Documentation*, 2019. dirección: https://registry.terraform.io/providers/terraform-provider-openstack/openstack/latest/docs.

[2] HashiCorp, *openstack*$_c$*ompute*$_i$*nstance*$_v$2, 2023. dirección: https://registry.terraform.io/providers/terraform-provider-openstack/openstack/latest/docs/resources/compute_instance_v2.

[3] HashiCorp, *openstack*$_c$*ompute*$_k$*eypair*$_v$2, 2023. dirección: https://registry.terraform.io/providers/terraform-provider-openstack/openstack/latest/docs/resources/compute_keypair_v2.

[4] HashiCorp, *Variable Definition Precedence*, 2023. dirección: https://developer.hashicorp.com/terraform/language/values/variables#variable-definition-precedence.

[5] HashiCorp, *Modules*, 2023. dirección: https://developer.hashicorp.com/terraform/language/modules.

[6] Omkar Birade, *Modules*, 2022. dirección: https://spacelift.io/blog/terraform-tfvars.

[7] Openstack, *Load Balancer as a Service (LBaaS)*, 2019. dirección: https://docs.openstack.org/ocata/networking-guide/config-lbaas.html.