Documento que acompaña a la práctica 2.

Experto Universitario DevOps&Cloud

Antonio Sánchez Antón

Información General

Propietario:	Antonio Sánchez	**Versión:**	1.0
Creado:	20/02/2021	**Actualizado:**	21/02/2021
Revisado para versión:		**Dirigido a:**	Profesores UNIR

Control de versiones

Versión	Acción	Descripción	Fecha
1.0	Nuevo	Versión inicial	20/02/2021

TABLA DE CONTENIDOS

1 INTRODUCCIÓN

El objetivo de la práctica es desplegar, en una serie de máquinas virtuales Linux, un clúster de Kubernetes y sobre él, una aplicación.

Las máquinas virtuales Linux se desplegarán en Azure.

Como aplicación he elegido

- https://hub.docker.com/r/alexwhen/docker-2048

Para alcanzar el objetivo de la práctica se hace uso de las herramientas

- Terraform:
  https://www.terraform.io/
  con la que se hace la creación de la infraestructura en Azure
- Ansible:
  https://www.ansible.com/
  con la que se automatiza toda la configuración de dicha infraestructura, así como, el despliegue de la aplicación.

Todo el código está disponible en:

https://github.com/asanton/Practica2Unir

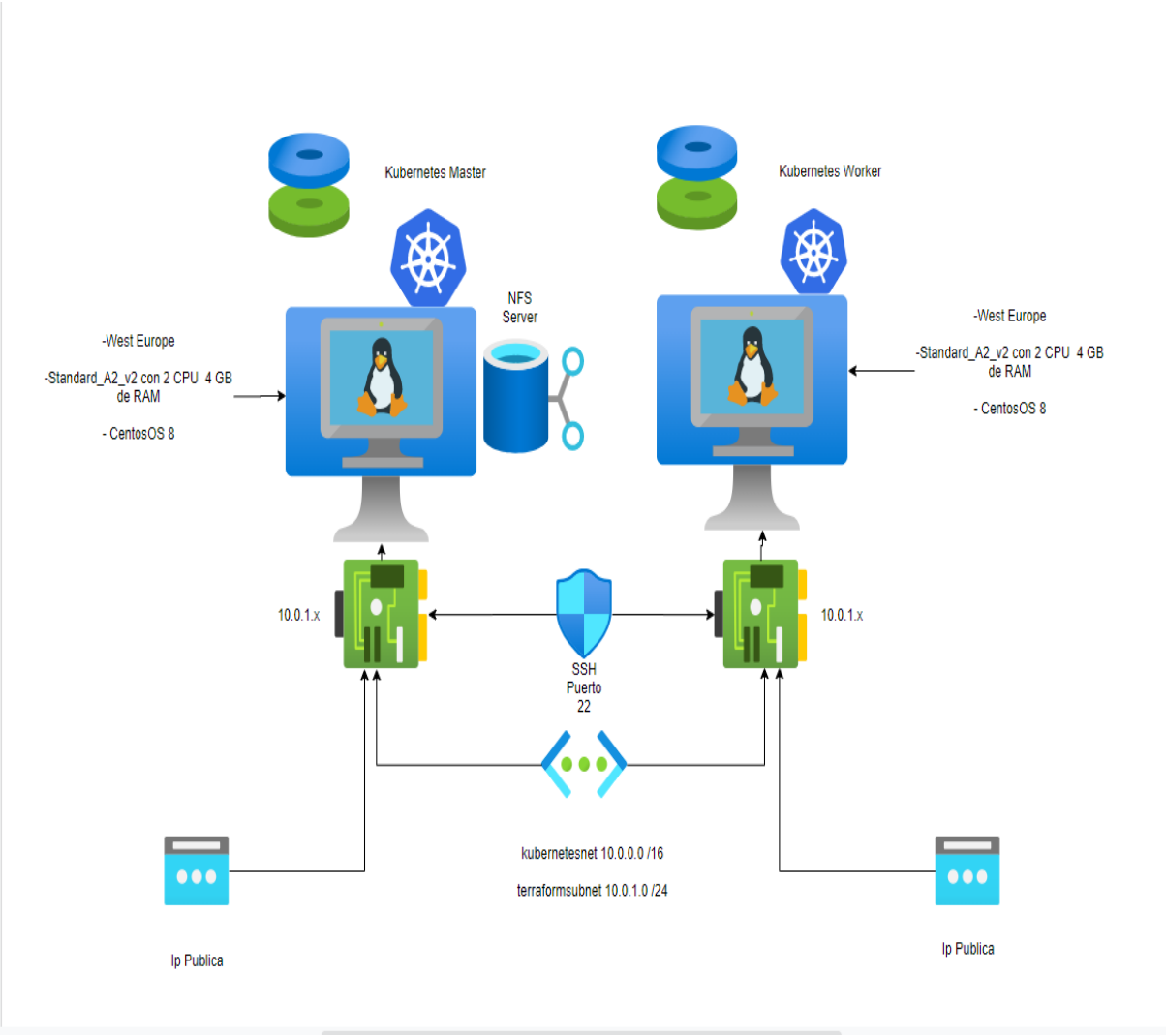
Se ha utilizado una maquina local Linux (CentOS 8), que será desde donde se lancen todas las automatizaciones, tanto Terraform como Ansible (Ansible Controller).

La práctica se ha desarrollado y verificado en local (maquinas Linux CentOs8 virtualizadas localmente con Virtual Box) como paso previo a su despliegue en Azure.

2 INFRAESTRUCTURA DESPLEGADA

Como se ha indicado anteriormente se hace uso de Terraform para desplegar la infraestructura necesaria en Azure.

Dicha infraestructura está formada por:



2 máquinas virtuales de tipo “Standard_A2_v2” con 2 CPU y 4Gb de RAM

Sobre ellas se despliega un sistema operativo Linux CentOS 8

Se crea una red “kubernetesnet” 10.0.0.0 /16 y sobre ella, la subnet “terraformsubnet” 10.0.1.0 /24 que será a la que pertenezcan las máquinas virtuales a desplegar.

Se crea un grupo de seguridad para ambas maquina donde, inicialmente, llevará una regla para permitir el acceso al puerto 22 (SSH).

Posteriormente, una vez desplegada la aplicación, se deberá abrir el puerto donde se ofrezca dicha aplicación

Una de las máquinas virtuales desempeñará los roles de máster de kubernetes y de servidor NFS y la otra desempeñará el rol de worker de kubernetes.

3 DESPLIEGUE REALIZADO

Si se requiere la preparación de una máquina desde la que hacer el despliegue revisar el “ANEXO: PREPARACIÓN MÁQUINA LOCAL”.

Todos los planes de Terraform y playbooks de Ansible están disponibles en:

https://github.com/asanton/Practica2Unir.git

Una vez clonado dicho repositorio tendremos 2 carpetas diferenciadas:

- Carpeta terraform

  Contiene los planes de Terraform para crear la infraestructura en Azure.

- Carpeta ansible

  Contiene los playbooks de Ansible con todas las tareas necesarias para configurar las maquinas.

Creación Infraestructura Azure

Nos posicionamos en la carpeta terraform.

Hay que editar el archivo de plantilla main.tf y configurar las credenciales de terraform:

```
provider "azurerm" {
 features {}
 subscription_id = "<SUBSCRIPCION ID>"
 client_id       = "<APP_ID>"
 client_secret   = "<PASSWORD>"
 tenant_id       = "<TENANT>"
}
```

Una vez realizado, ejecutar los comandos:

-terraform init

-terraform apply

La infraestructura se creará.

Una vez creada hay que:

a) Identificar las direcciones ip de las maquinas creadas.

b) Conectar via ssh, con el usuario “adminUsername”, a las máquinas creadas para verificar que el despliegue es correcto.

Configuración infraestructura y despliegue de la aplicación

He utilizado diferentes módulos de ansible para realizar las tareas de configuración necesarias (command, shell, user, file, dnf, lineinfile …).

He utilizado “ansible-galaxy” para crear los roles a utilizar:

- “antoniounir.configuracioncomun”

  Se encarga de realizar todas las tareas de configuración comunes a todas las máquinas del inventario

- “antoniounir.configuracionNFS”

  Se encarga de realizar todas las tareas necesarias para dejan configurado un NFS Server

- “antoniounir.configuracionK8sMasterWorkers”

  Se encarga de realizar todas las tareas comunes a los nodos máster y workers de un clúster de Kubernetes

- “antoniounir.configuracionK8sMaster”

  Se encarga de realizar todas las tareas necesarias para dejan configurado el nodo máster de un clúster de Kubernetes

- “antoniounir.configuracionK8sWorkers”

  Se encarga de realizar todas las tareas necesarias para dejan configurado el nodo / los nodos worker de un clúster de Kubernetes

- “antoniounir.despliegueAppK8s”

  Se encarga de realizar todas las tareas necesarias para desplegar una aplicación en el clúster de Kubernetes

Nos posicionamos en la carpeta ansible donde tenemos:

- inventario_local.yml

  Como comenté, realicé la práctica en local como paso previo al despliegue en Azure.

  Este fichero contiene el inventario local.

- inventario_azure.yml

  Inventario desplegado en Azure.

- playbookDespliegueApp.yml

  Tareas que se encargan de desplegar la aplicación en el clúster de Kubernetes.

- playbookDespliegueK8s.yml

  Tareas que se encargan de desplegar un clúster de kubernetes con un nodo master, un servidor NFS y tantos workers como se indiquen en el inventario.
  Al finalizar el clúster esta levantado y los workers correctamente unidos al mismo.

- playbookPrueba.yml

  Utilizado internamente para probar el funcionamiento de los módulos de Ansible a utilizar.

Para crear el clúster de Kubernetes (con el nodo máster, NFS server y workers) hay que ejecutar el comando:

ansible-playbook -i inventario_azure.yml playbookDespliegueK8s.yml

Para desplegar en dicho clúster la aplicación hay que ejecutar el comando:

ansible-playbook -i inventario_azure.yml playbookDespliegueApp.yml

Una vez el playbook termine, nos conectamos a la dirección ip de la máquina que hace de máster de Kubernetes y ejecutamos el comando:

kubectl get svc -n haproxy-controller

fijándonos en la columna “PORT(S)” para saber el puerto que nos sirve la aplicación

PORT(S)

*80:**31741**/TCP,443:30847/TCP,1024:32078/TCP*

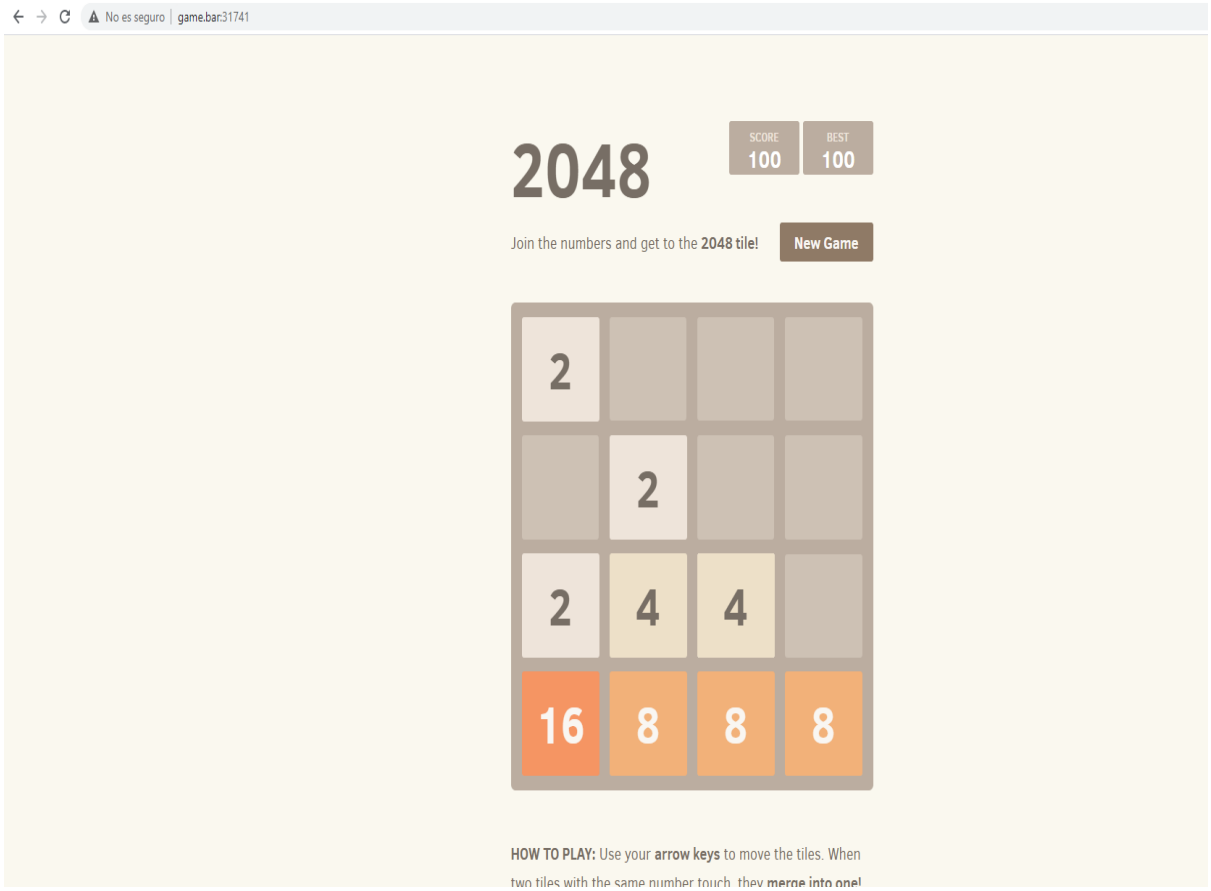
Con este puerto, en caso de Azure, hay que abrir una regla de entrada en el grupo de seguridad, similar a la que tenemos para SSH puerto 22, pero para HTTP puerto 31741.

Una vez todo este realizado, desde un navegador introducimos la url:

http://game.bar:31741

y accederemos a la aplicación (Importante poder resolver el nombre “game.bar” a la dirección ip del nodo máster).

Esta parte, debido a un problema en con Calico en Azure, solo se ha podido verificar en local


No es seguro | game.bar:31741
2048
SCORE
100
BEST
100
Join the numbers and get to the 2048 tile!
New Game
2
2
2 4 4
16 8 8 8
HOW TO PLAY: Use your arrow keys to move the tiles. When
two tiles with the same number touch, they merge into one!

4 PROBLEMAS ENCONTRADOS

Fundamentalmente he encontrado 3 problemas que me han retrasado

a) Problema con la cuenta “Azure for Students” que me impedía crear un “StorageAccount”

Al crear el plan de Terraform da un error:

"Error: Error reading queue properties for AzureRM Storage Account.....Code= AutenticationFailed"

Dicho error esta notificado a los profesores de la asignatura y, en el momento actual, no hay solución.

Como “workaround” siguiendo las indicaciones de los profesores se crea el plan sin hacer uso del “StorageAccount”.

b) Problema en el despliegue de Kubernetes

La infraestructura original para desplegar estaba formada por 4 máquinas Linux:

- Kubernetes Máster
- Kubernetes Worker1
- Kubernetes Worker2
- NFS Server

Cada máquina con 1 CPU y 3,5 GB de memoria RAM, dimensionamiento que era compatible con la limitación a 4 CPU que tiene la cuenta “Azure for Students”.

Al desplegar el Máster de Kubernetes tuve problemas, ya que el playbook no daba error alguno, pero no podía unir ningún worker al clúster.

Tras múltiples pruebas e investigaciones, ejecutando a mano la inicialización de kubeadm con la opción -v=5 obtuve el error:

[ERROR NumCPU]: the number of available CPUs 1 is less than the required 2

El máster de kubernetes necesita como poco 2 CPU, lo que me obliga a replantear la infraestructura a desplegar, infraestructura que estará finalmente formada por 2 máquinas Linux:

- Kubernetes Máster y NFS Server
- Kubernetes Worker1

Cada máquina con 2 CPU y 4 Gb de memoria RAM, dimensionamiento compatible con la limitación a 4 CPU que tiene la cuenta “Azure for Students”.

5 ANEXO: PREPARACION MAQUINA LOCAL

Como he comentado, todo se lanzará desde una maquina local Linux (CentOS 8) de la cual tengo acceso “root”.

La preparación de dicha maquina consiste en los siguientes elementos:

a) Actualización de dicha maquina (conectado como usuario “root”)

dnf update -y

b) Crear un usuario “ansible” en las maquinas a gestionar (Este paso se omite para despliegue Azure, ya que ya existe un usuario “adminUsername”) (conectado como usuario “root”)

useradd -md /home/ansible ansible
passwd ansible
usermod -aG wheel ansible --> privilegios sudo a ansible
usermod -s /bin/bash ansible
sudo visudo --> Descomentar lineas %wheel ALL=(ALL) PASSWD:ALL

c) Generar clave ssh para este usuario y distribuirlas a las máquinas a gestionar (Este paso se omite para despliegue Azure, ya que ya existe) (conectado como usuario creado en punto anterior, es decir, “ansible”)

ssh-keygen
ssh-copy-id ansible@<ip maquina kubernetesMaster>
ssh-copy-id ansible@<ip maquina kubernetesNodo1>

d) Instalar Ansible, Terraform y GIT en Ansible Controller (conectado como usuario “ansible”)

Para ello:

d1) Actualizar cache dnf

sudo dnf makecache

d2) Instalar Git

sudo dnf install git -y

d3) Activar EPEL repositorio

sudo dnf install epel-release -y

d4) Actualizar cache dnf

sudo dnf makecache -y

d5) Instalar Ansible

sudo dnf install ansible -y

d6) Comprobar instalación

ansible --version

d7) Añadir al host del Ansible Controler mapeo ip dominio para las maquinas a gestionar (Este paso se omite para despliegue en Azure)

sudo vi /etc/hosts

d8) Instalar wget

sudo dnf install wget -y

d9) Instalar terraform

sudo wget *https://releases.hashicorp.com/terraform/0.14.6/terraform_0.14.6_linux_amd64.zip*

sudo unzip ./terraform_0.14.6_linux_amd64.zip -d /usr/local/bin

terraform -version

d10) Clonamos el repositorioGIT

sudo git clone https://github.com/asanton/Practica2Unir.git

Tras estos pasos dispondremos de 2 carpetas:

- terraform
  Donde están las plantillas terraform para desplegar la infraestructura
- ansible
  Donde están los playbooks para realizar las configuraciones necesarias para montar el clúster de kubernetes y desplegar la aplicación