

Resumen

Breve descripción del proyecto

Objetivos principales

Introducción

Contexto y justificación del proyecto

Alcance del proyecto

Metodología utilizada

Arquitectura del sistema

Descripción general de la arquitectura

Componentes principales

Relaciones y comunicaciones entre los componentes

Entorno local

Herramientas utilizadas para el desarrollo local

Configuración del entorno local

Despliegue de la máquina virtual

Descripción de la máquina virtual y sus características

Configuración de la VM

Configuración del servidor Apache

Puesta en marcha del servidor contenerizado

Consideraciones para el despliegue

Despliegue de AKS

Descripción de Azure Kubernetes Service (AKS)

Descripción del frontend (AzureVote) y backend (Redis)

Customización de Imágenes, Push al Registry y Despl. de Pods

Consideraciones para el despliegue en AKS

Infraestructura como código con Terraform

Descripción de Terraform y su papel en el proyecto

Estructura del código de infraestructura

Organización y estructura del código de Terraform

Configuración con Ansible

Descripción de Ansible y su papel en el proyecto

Organización y estructura del código de Ansible

Automatización de la configuración del sistema

Proceso de despliegue automatizado

Scripts de despliegue automatizado

Pruebas y validación

Estrategia de pruebas utilizada

Monitorización y registro

Anexos

Diagramas de arquitectura

Capturas de pantalla

Lecciones Aprendidas

Conocimientos técnicos adquiridos

Comprensión teórica alcanzada

Experiencia práctica obtenida

Desafíos

Gestión de la configuración y errores

Licencia

La elección de la licencia MIT

1. Resumen

El presente informe describe como realizar una automatización de despliegues en entornos Cloud. El objetivo principal de esta actividad es crear infraestructura de forma automatizada en un proveedor de Cloud pública, utilizando herramientas de gestión de la configuración para automatizar la instalación y configuración de servicios.

1.1 Breve descripción del proyecto:

El caso práctico 2 consiste en desplegar un repositorio de imágenes de contenedores Azure Container Registry (ACR), una aplicación en forma de contenedor utilizando Podman sobre una máquina virtual en Azure, un cluster de Kubernetes como servicio gestionado en Microsoft Azure (AKS), y una aplicación con almacenamiento persistente sobre el cluster AKS.

1.2 Objetivos principales:

Crear infraestructura de forma automatizada en un proveedor de Cloud pública.

Utilizar herramientas de gestión de la configuración para automatizar la instalación y configuración de servicios.

Desplegar mediante un enfoque totalmente automatizado aplicaciones en forma de contenedor sobre el sistema operativo.

Desplegar mediante un enfoque totalmente automatizado aplicaciones que hagan uso de almacenamiento persistente sobre una plataforma de orquestación de contenedores.

2.Introducción

La automatización de despliegues en entornos Cloud se ha vuelto fundamental en el desarrollo de aplicaciones y la gestión de infraestructuras. Este informe presenta el caso práctico 2, que se centra en el despliegue automatizado de aplicaciones en Microsoft Azure utilizando herramientas como Terraform y Ansible. El objetivo es proporcionar una infraestructura escalable y confiable, y facilitar el despliegue de aplicaciones en contenedores.

2.1 Contexto y justificación del proyecto

En un entorno cada vez más dinámico y exigente, la automatización de despliegues se vuelve esencial para garantizar la eficiencia y la rapidez en la implementación de aplicaciones. Además, la adopción de herramientas de gestión de la configuración y prácticas de infraestructura como código mejora la consistencia, la reproducibilidad y la colaboración en el proceso de despliegue.

En este contexto, Microsoft Azure se posiciona como una plataforma líder en servicios de Cloud pública, ofreciendo un entorno confiable y escalable. El uso de Terraform como herramienta de infraestructura como código y Ansible como herramienta de gestión de la configuración proporciona una solución sólida y flexible para automatizar el despliegue y la configuración de servicios en Azure.

2.2 Alcance del proyecto

El alcance de este proyecto se centra en el despliegue automatizado de un repositorio de imágenes de contenedores utilizando Azure Container Registry (ACR), una aplicación en forma de contenedor utilizando Podman sobre una máquina virtual en Azure y un cluster de Kubernetes gestionado en Azure (AKS). También se incluye la implementación de una aplicación con almacenamiento persistente sobre el cluster AKS.

La automatización se realizará mediante el uso de Terraform para la creación de la infraestructura y Ansible para la configuración de los servicios. Se han evitado el uso de módulos Command, Shell y Script en Ansible, y buscando utilizar las funcionalidades nativas de las herramientas.

2.3 Metodología utilizada

Para llevar a cabo este proyecto, se ha seguido un enfoque ágil basado en iteraciones y entregas incrementales. Para el registro de las mismas, se ha utilizado la herramienta de control de versiones Git, para gestionar el código fuente y facilitar el avance del proyecto y la disponibilidad del código.

3.Arquitectura del sistema:

Servicio de Kubernetes (aks_cluster_kubernetes)

Conjunto de escalas de máquina virtual (aks-agentpool-26279739-vmss)

Identidad administrada (aks_cluster_kubernetes-agentpool)

Equilibrador de carga (kubernetes)

Red virtual (aks-vnet-28408381, vnet)

Registro de contenedor (maseiraacr)

Máquina virtual (podmanVM)

Disco Máquina virtual (podmanVM_disk1_a5be0f43fae447dfb3bc8f0743c0248e)

Interfaz de red Máquina virtual (podmanVM_nic)

Dirección IP pública Máquina virtual (podmanVM_public_ip)

Dirección IP pública para el frontend del AKS (7fe98094-5ac5-4f8f-8c5b-de1de0b9c97a)

Grupo de seguridad del aks (aks-agentpool-28408381-nsg)

Tabla de rutas del aks (aks-agentpool-28408381-routetable)

Network Watcher (NetworkWatcher_uksouth)

3.1 Descripción general de la arquitectura:

La arquitectura para el despliegue del servidor web contenerizado se centra en la máquina virtual podmanVM que se despliega en el grupo de recursos cp2_resource_group en la región UK South. Utiliza una dirección IP pública (7fe98094-5ac5-4f8f-8c5b-de1de0b9c97a) para la comunicación externa.

La arquitectura para el despliegue de un clúster de Kubernetes en Azure. El servicio de Kubernetes (aks_cluster_kubernetes) es el núcleo de la arquitectura y se despliega en el grupo de recursos cp2_resource_group en la región UK South. Utiliza una dirección IP pública (7fe98094-5ac5-4f8f-8c5b-de1de0b9c97a) para la comunicación externa.

3.2 Componentes principales:

vnet : Proporciona una red privada virtualizada para los recursos del cp2_resource_group

maseiraacr : Almacena y administra imágenes de contenedor utilizadas en el clúster de Kubernetes y podmanVM.

aks_cluster_kubernetes: Proporciona una plataforma para la implementación y administración de contenedores.

aks-agentpool-26279739-vmss: Permite escalar automáticamente los pods dentro del clúster de Kubernetes según demanda.

aks_cluster_kubernetes-agentpool: funcionalidad de Azure que permite a las aplicaciones y servicios acceder a los recursos de Azure de forma segura sin la necesidad de almacenar credenciales de autenticación. En el contexto de AKS, una identidad administrada es un objeto de Azure Active Directory (AAD) que se asigna a un clúster de AKS y permite que el clúster acceda a los recursos de Azure, como bases de datos o servicios de almacenamiento, de manera segura y controlada.

Kubernetes (Equilibrador de carga): Distribuye el tráfico de red entrante a los nodos del clúster de Kubernetes.

aks-vnet-28408381 : Proporciona una red privada virtualizada para los recursos del clúster de Kubernetes.

aks-agentpool-28408381-nsg: Controla el tráfico de red y aplica políticas de seguridad específicas en el clúster de Kubernetes.

aks-agentpool-28408381-routetable: Define las rutas de red utilizadas por los recursos del clúster de Kubernetes.

7fe98094-5ac5-4f8f-8c5b-de1de0b9c97a : Proporciona una dirección IP accesible públicamente para los pods de aks_cluster_kubernetes .

kubernetes-a64d40d9f9abe4ed0a01173058f1d93f : Proporciona una dirección IP accesible públicamente para los pods de aks_cluster_kubernetes .

podmanVM: Puede albergar otros componentes del sistema y proporcionar capacidad computacional adicional según sea necesario.

podmanVM_disk1_a5be0f43fae447dfb3bc8f0743c0248e : Almacena datos persistentes para podmanVM .

podmanVM_nic : Permite la comunicación de red entre podmanVM y los componentes del sistema.

podmanVM_public_ip : Proporciona una dirección IP accesible públicamente para podmanVM.

Network Watcher: Proporciona diagnósticos y monitoreo de red para la infraestructura de Azure.

3.3 Relaciones y comunicaciones entre los componentes:

El servicio de Kubernetes (aks_cluster_kubernetes) utiliza los conjuntos de escalas de máquina virtual para escalar automáticamente los recursos según la demanda.

Los nodos del clúster de Kubernetes se comunican entre sí y con otros componentes utilizando la red virtual (aks-vnet-28408381, vnet).

El equilibrador de carga (kubernetes) distribuye el tráfico de red entrante a los nodos del clúster de Kubernetes.

La identidad administrada (aks_cluster_kubernetes-agentpool) proporciona una identidad segura para los componentes del clúster de Kubernetes.

El grupo de seguridad de red (aks-agentpool-28408381-nsg) controla el tráfico de red y aplica políticas de seguridad en el clúster de Kubernetes.

La tabla de rutas (aks-agentpool-28408381-routetable) define las rutas de red utilizadas por los recursos del clúster de Kubernetes.

El registro de contenedor (maseiraacr) almacena y administra las imágenes de contenedor utilizadas en el clúster de Kubernetes.

La máquina virtual (podmanVM) puede interactuar con otros componentes a través de la interfaz de red (podmanVM_nic) y la dirección IP pública (podmanVM_public_ip) y usa como almacenamiento persistente podmanVM_disk1_a5be0f43fae447dfb3bc8f0743c0248e.

El Network Watcher (NetworkWatcher_uksouth) proporciona diagnósticos y monitoreo de red para la infraestructura de Azure.

4.Entorno local

Como módulo de control y operación he optado por mi maquina local W11 con WLS empleando una imagen de Ubuntu 22.04.2 LTS.

Para alojar todos los archivos de configuración del caso práctico se ha empleado el siguiente repo de GITHUB:

<https://github.com/OAlvarezOliveira/casopractico2.git>

Se valoró la posibilidad de emplear una máquina virtual alojada en AWS para maximizar el uso de la cuenta de estudiante de la práctica 1, esta opción se descartó por qué puede aumentar la configuración de la conectividad y la latencia en ciertos escenarios.

4.1 Herramientas utilizadas para el desarrollo local

Infraestructura como código: Terraform v1.5.2

- Instala el paquete terraform

```
sudo apt install terraform --classic
```

Monitoreo y diagnóstico: Azure CLI (Command Line Interface) 2.49.0

- Agregar la clave GPG del repositorio de paquetes de Microsoft

```
curl -sL https://packages.microsoft.com/keys/microsoft.asc | sudo gpg --dearmor -o  
/etc/apt/trusted.gpg.d/microsoft.gpg
```

- Agregar el repositorio de paquetes de Microsoft a mi lista de fuentes de apt

```
echo "deb [arch=amd64] https://packages.microsoft.com/repos/azure-cli/  
$(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/azure-cli.list
```

- Actualizar el índice de paquetes apt

```
sudo apt update
```

- Instala el paquete azure-cli

```
sudo apt install azure-cli
```

Orquestación e integración con la nube: ansible 2.10.8

- Instala el paquete ansible

```
sudo apt install ansible
```

4.2 Configuración del entorno local

Para el correcto desempeño del 03_playbook.yml encargado de adjuntar el ACR creado al AKS se deben crear las variables de entorno AZURE_USERNAME, AZURE_PASSWORD y AZURE_TENANT

para el nombre de usuario, la contraseña y el tenant de Azure, respectivamente.

5.Despliegue de la máquina virtual

El despliegue de la máquina virtual se lleva a cabo por medio de terraform quedando las características de la misma definidas en recursos.tf

5.1 Descripción de la máquina virtual y sus características

La configuración de Terraform incluida en recursos.tf define una máquina virtual en Azure utilizando el recurso `azurerm_linux_virtual_machine`.

La descripción de la máquina virtual y sus características es la siguiente :

Nombre: "podmanVM"

Ubicación: La ubicación es especificada por la variable `var.location_name`.

Grupo de recursos: El nombre del grupo de recursos se obtiene de la variable `var.resource_group_name`.

Tamaño: "Standard_DS1_v2". 1 núcleo virtual (vCPU), 3.5 GB de memoria RAM y 50 GB de almacenamiento temporal basado en unidades de estado sólido (SSD). El almacenamiento temporal es una memoria rápida y de alto rendimiento que se utiliza para operaciones temporales y caché. No es persistente y los datos almacenados en esta unidad se pierden cuando la máquina virtual se detiene o se reinicia.

Nombre de usuario administrativo: "adminuser". Este es el nombre de usuario para acceder a la máquina virtual.

Clave SSH: La clave pública SSH se especifica utilizando el archivo `"/home/server_admin/.ssh/id_rsa.pub"`. Esto permite la autenticación mediante SSH con el usuario administrativo especificado.

Interfaces de red: La máquina virtual se conecta a la red mediante la interfaz de red `azurerm_network_interface.nic_podman_vm`.

Almacenamiento permanente: La máquina virtual tiene un disco de sistema operativo con las siguientes características:

Caché: "ReadWrite". Esto especifica que el disco permite operaciones de lectura y escritura en caché.

Tipo de cuenta de almacenamiento: "Standard_LRS". Esto indica que el disco se almacena de manera duradera en Azure Storage utilizando el nivel de servicio estándar.

Imagen de origen: La máquina virtual utiliza una imagen de referencia de CentOS 8 proporcionada por el editor "cognosys". La SKU de la imagen es "centos-8-stream-free", y la versión es "22.03.28".

Plan: El plan de la máquina virtual especifica los detalles del producto y el editor de la imagen.

Esta configuración crea una máquina virtual en Azure utilizando CentOS 8 como sistema operativo, con una interfaz de red que se conecta a una subred en una red virtual y una dirección IP pública estática asignada.

Los detalles específicos, como la ubicación, el nombre del grupo de recursos se proporcionan desde el archivo vars.tf para Terraform.

5.2 Configuración de la VM

La configuración de podmanVM la llevaremos a cabo por medio de 00_playbook.yml , que se compone de las siguientes tareas:

Tarea 1 - Instalar herramientas requeridas

Utiliza el módulo ansible.builtin.yum para instalar los siguientes paquetes en la VM: podman, skopeo, httpd, httpd-tools y openssl.

Tarea 2 - Instalar el módulo passlib

Utiliza el módulo ansible.builtin.pip para instalar el módulo passlib a través de pip.

Tarea 3 - Crear directorio de trabajo

Utiliza el módulo `ansible.builtin.file` para crear un directorio llamado `"/webserver"` en la VM. Se especifican los permisos de propietario, grupo y modo del directorio.

Tarea 4 - Crear archivo de credenciales

Utiliza el módulo `community.general.htpasswd` para crear un archivo de credenciales llamado `"/webserver/.cred"`. Se especifica un nombre de usuario ("caramelo") y una contraseña ("mentolin") para el archivo.

Tarea 5 - Generar certificado autofirmado

Utiliza el módulo `community.crypto.openssl_privatekey` para generar una clave privada de OpenSSL en el archivo `"/webserver/localhost.key"` con un tamaño de 2048 bits.

Tarea 6 - Crear petición de firma del certificado: Utiliza el módulo `community.crypto.openssl_csr` para generar una solicitud de firma de certificado (CSR) en el archivo `"/webserver/localhost.csr"`. Se especifican los detalles del sujeto del certificado, como el país, estado, localidad, organización, unidad organizativa y nombre común.

Tarea 7 - Crear certificado autofirmado:

Utiliza el módulo `community.crypto.x509_certificate` para generar un certificado autofirmado en el archivo `"/webserver/localhost.crt"` utilizando la clave privada y la CSR generadas en las tareas anteriores. Se especifica que el certificado es autofirmado y se establecen los permisos del archivo.

Estas tareas en su conjunto permiten configurar la VM con las herramientas necesarias, crear un directorio de trabajo, generar un archivo de credenciales y crear un certificado autofirmado para su uso en un servidor web.

Permisos del directorio: los permisos del directorio establecidos `"0700"` asegura que solo el propietario pueda acceder al directorio y ver su contenido. Los otros usuarios y el grupo no tendrán acceso al directorio en sí.

Permisos de los archivos sensibles: los permisos `"0664"` en los archivos individuales que contienen información sensible aseguran que solo el propietario pueda leer o modificar los archivos, y otros

usuarios solo tendrán acceso de lectura a su contenido.

En cuanto al localhost.crt asignamos "0600" que garantiza que solo el propietario tenga acceso completo al archivo, mientras que el grupo y otros usuarios no tienen permisos para leer o modificar su contenido puesto que es el certificado del servidor.

En general, estos permisos defienden los archivos y el directorio al establecer restricciones de acceso apropiadas según mi criterio y experiencia profesional.

5.3 Configuración del servidor Apache

Para la configuración de la imagen del servidor Apache se ha creado el archivo 01_playbook.yml realiza el siguiente desglose de tareas:

Tarea 8 - Crear archivo index.html

Crea un archivo "index.html" utilizando una plantilla definida en "templates/index.html". El archivo resultante se coloca en "/webserver/index.html".

Tarea 9 - Crear archivo httpd.conf

Crea un archivo "httpd.conf" utilizando una plantilla definida en "templates/httpd.conf". El archivo resultante se coloca en "/webserver/httpd.conf".

Tarea 9 - Crear archivo .htaccess

Crea un archivo ".htaccess" utilizando una plantilla definida en "templates/.htaccess". El archivo resultante se coloca en "/webserver/.htaccess".

Tarea 10 - Crear archivo Containerfile

Crea un archivo "Containerfile" utilizando una plantilla definida en "templates/Containerfile".

La opción "become: true" se usa para ejecutar las tareas como el usuario "root".

Una vez que la imagen se construye y se suba al ACR, los permisos y propietarios de los archivos creados por este playbook no tienen relevancia. Incluso sería aconsejable su eliminación del servidor que aloja el Apache.

5.3 Puesta en marcha del servidor contenerizado

Como último paso para desplegar nuestro servidor web se ha creado el 02_playbook.yml que consta de las siguientes tareas:

Tarea 12 - Generar imagen del contenedor

Esta tarea construye una imagen de contenedor utilizando Podman con la etiqueta especificada (webserver) y el Containerfile ubicado en /webserver/Containerfile.

Tarea 13 - Etiquetar imagen del contenedor

Esta tarea etiqueta la imagen construida anteriormente (localhost/webserver:latest) con una nueva etiqueta ({{ registry_acr }}/webserver:casopractico2) utilizando el módulo containers.podman.podman_tag.

Tarea 14 - Iniciar sesión en el registro de contenedores

Esta tarea inicia sesión en un registro de contenedores ({{ registry_acr }}) utilizando el nombre de usuario especificado ({{ registry_username }}) y la contraseña ({{ registry_password }}) con el módulo containers.podman.podman_login.

Tarea 15 - Subir imagen del contenedor al registro

Esta tarea envía la imagen del contenedor ({{ registry_acr }}/webserver:casopractico2) al registro de contenedores utilizando el módulo containers.podman.podman_image con la opción push establecida en yes.

Tarea 16 - Crear contenedor

Esta tarea crea un contenedor llamado web utilizando la imagen ({{ registry_acr }}/webserver:casopractico2) y mapea el puerto 8080 del host al puerto 443 del contenedor.

Tarea 17 - Generar los archivos de systemd para el contenedor

Esta tarea genera archivos de unidad de systemd para administrar el contenedor utilizando el comando podman generate systemd.

Tarea 18 - Copiar los archivos generados al directorio de systemd

Esta tarea copia el archivo de unidad systemd generado (container-web.service) al directorio /etc/systemd/system/.

Tarea 19 - Recargar la configuración de systemd

Esta tarea recarga la configuración de systemd para incluir el nuevo archivo de unidad utilizando el módulo `ansible.builtin.systemd` con la opción `daemon_reload` establecida en `yes`.

Tarea 20 - Habilitar y activar el servicio del contenedor

Esta tarea habilita y pone en marcha el servicio de systemd para el contenedor (`container-web.service`) utilizando el módulo `ansible.builtin.systemd` con la opción `enabled` establecida en `yes` y la opción `state` establecida en `started`.

En resumen, este playbook construye una imagen de contenedor, la envía a un registro, crea un contenedor y lo configura como un servicio de systemd para administrar el servidor containerizado.

5.4 Consideraciones para el despliegue

Para el despliegue con Ansible debe estar creados los siguientes elementos:

`/templates` : este directorio aloja los archivos `.htaccess` , `http.conf` , `index.html` y `Containerfile`.

- `.htaccess`: es un archivo de configuración utilizado por el servidor web Apache. Se coloca en un directorio específico del sitio web y contiene directivas que afectan al comportamiento del servidor en ese directorio y sus subdirectorios. Puede utilizarse para configurar la autenticación, redirecciones, reescritura de URL y otras reglas de acceso y seguridad.

Explicación de cada directiva:

- `AuthType Basic`: Esta directiva especifica el tipo de autenticación a utilizar, que en este caso es la autenticación básica. La autenticación básica envía las credenciales de usuario y contraseña en texto plano a través del encabezado de la solicitud HTTP. Es importante tener en cuenta que la autenticación básica no cifra las credenciales, por lo que se recomienda utilizar HTTPS para proteger la seguridad de la información de inicio de sesión.
- `AuthName "Restricted Content"`: Esta directiva define el mensaje que se muestra en el cuadro de diálogo de autenticación cuando se requiere iniciar sesión. En este caso, se muestra el mensaje "Restricted Content" como indicación de que el acceso está restringido y se requiere autenticación.

- AuthUserFile /usr/local/apache2/.cred: Esta directiva especifica la ubicación del archivo de contraseñas que contiene las credenciales de los usuarios permitidos. En este caso, el archivo se encuentra en el directorio "/usr/local/apache2/" y se llama ".cred". Es importante asegurarse de que este archivo esté correctamente configurado con las credenciales de usuario y contraseña adecuadas.
- Require valid-user: Esta directiva establece que solo los usuarios válidos que han iniciado sesión correctamente pueden acceder al contenido protegido. Esto significa que cualquier usuario que proporcione credenciales de inicio de sesión válidas podrá acceder al contenido.
- Containerfile: es un archivo de texto utilizado para definir los pasos necesarios para construir una imagen de contenedor. Es similar a un Dockerfile en el contexto de Docker. En este caso, el Containerfile se utiliza en conjunto con la herramienta Podman para construir una imagen de contenedor utilizando las instrucciones y comandos definidos en el archivo.

Explicación de cada instrucción:

- FROM docker.io/httpd:latest: Esta instrucción establece la imagen base para la construcción del contenedor. En este caso, se utiliza la imagen docker.io/httpd:latest, que es la imagen más reciente de Apache HTTP Server disponible en el repositorio Docker.
- COPY index.html /usr/local/apache2/htdocs/: Esta instrucción copia el archivo index.html al directorio /usr/local/apache2/htdocs/ dentro del contenedor. Este archivo se utiliza como la página principal del sitio web.
- COPY .htaccess /usr/local/apache2/htdocs/: Esta instrucción copia el archivo .htaccess al directorio /usr/local/apache2/htdocs/ dentro del contenedor. Este archivo contiene las directivas de configuración para la autenticación básica, como se mencionó anteriormente.
- COPY httpd.conf /usr/local/apache2/conf/: Esta instrucción copia el archivo httpd.conf al directorio /usr/local/apache2/conf/ dentro del contenedor. Este

archivo es la configuración principal del servidor web Apache y se utiliza para personalizar y ajustar el comportamiento del servidor.

- COPY .cred /usr/local/apache2/: Esta instrucción copia el archivo .cred al directorio /usr/local/apache2/ dentro del contenedor. Este archivo contiene las credenciales de usuario utilizadas para la autenticación básica, según se referenciaba en el archivo .htaccess.
 - COPY localhost.key /usr/local/apache2/: Esta instrucción copia el archivo localhost.key al directorio /usr/local/apache2/ dentro del contenedor. Este archivo es una clave privada utilizada para configurar HTTPS en el servidor web Apache.
 - COPY localhost.crt /usr/local/apache2/: Esta instrucción copia el archivo localhost.crt al directorio /usr/local/apache2/ dentro del contenedor. Este archivo es un certificado SSL utilizado para configurar HTTPS en el servidor web Apache.
- httpd.conf: El archivo httpd.conf es el archivo principal de configuración del servidor web Apache. Contiene directivas que definen cómo se comporta el servidor web, como la configuración de los directorios raíz, los módulos a cargar, los ajustes de seguridad y otras opciones de configuración global del servidor.

Explicación de cada directiva:

- ServerRoot: Esta directiva especifica la ubicación del directorio raíz del servidor Apache.
- Listen: Esta directiva especifica el puerto en el que el servidor Apache escucha las solicitudes entrantes. En este caso, se establece en el puerto 443, que es el puerto estándar para HTTPS.
- LoadModule: Estas directivas cargan módulos adicionales de Apache necesarios para habilitar diversas funcionalidades. Cada línea de LoadModule carga un módulo específico.
- User y Group: Estas directivas especifican el usuario y el grupo bajo los cuales se ejecutará el servidor Apache.
- ServerAdmin: Esta directiva establece la dirección de correo electrónico del administrador del servidor.

- ServerName: Esta directiva establece el nombre del servidor.
 - SSLEngine, SSLCertificateFile, SSLCertificateKeyFile: Estas directivas habilitan y configuran la funcionalidad de SSL/TLS para el servidor Apache, especificando la ubicación del certificado y la clave privada utilizados para la comunicación segura.
 - <Directory>: Estas directivas definen las configuraciones y permisos para diferentes directorios del servidor. En este caso, se especifican la configuración para el directorio raíz y el directorio de documentos (/usr/local/apache2/htdocs).
 - DirectoryIndex: Esta directiva especifica el nombre del archivo que se servirá automáticamente cuando se acceda a un directorio sin especificar un nombre de archivo específico. En este caso, se establece en index.html.
 - <Files>: Esta directiva establece las restricciones de acceso a archivos con nombres que comienzan con ".ht". En este caso, se niega el acceso a esos archivos.
 - ErrorLog: Esta directiva especifica la ubicación del archivo de registro de errores del servidor Apache.
 - LogLevel: Esta directiva establece el nivel de registro para los mensajes de registro del servidor Apache.
 - ScriptAlias: Esta directiva establece la ubicación del directorio de scripts CGI.
 - AddType: Estas directivas definen tipos MIME adicionales para ciertas extensiones de archivo.
 - <IfModule>: Estas directivas envuelven secciones de configuración que se aplican solo si los módulos correspondientes están cargados.
- index.html: El archivo index.html es el archivo principal de un sitio web.

También se debe añadir la dirección IP pública de podmanVM al archivo hosts.txt, en este archivo, aún que no es necesario, debido a algunos errores durante el desarrollo del caso, se ha añadido a este archivo los siguientes parámetros:

`ansible_user=adminuser`: Esta línea indica el nombre de usuario utilizado para conectarse al host. En este caso, se establece como "adminuser".

`ansible_ssh_private_key_file=/home/server_admin/.ssh/id_rsa`: Esta línea especifica la ubicación del archivo de clave privada SSH utilizada para la autenticación en el host. En este caso, se establece como "/home/server_admin/.ssh/id_rsa".

`ansible_python_interpreter=/usr/bin/python3`: Esta línea indica la ubicación del intérprete de Python en el host. En este caso, se establece como "/usr/bin/python3".

Por último en el mismo directorio debe existir un archivo `vars.yaml` que entre otras variables almacenará nuestro

`registry_acr`: nombre del ACR creado

`registry_username`: nombre del usuario del ACR

`registry_password`: contraseña del registro

`kubernetes_cluster_name`: nombre del clúster de AKS

`resource_group_name`: nombre del grupo de recursos

6. Despliegue de AKS

El despliegue del AKS se lleva a cabo por medio de terraform quedando las características de la misma definidas en `recursos.tf`

6.1 Descripción de Azure Kubernetes Service (AKS)

El tamaño mínimo de máquina virtual (nodo en AKS) aceptado debe tener al menos 2 cores con 4GB de memoria. La selección por defecto son máquinas Linux con tamaño `Standard_DS2_v2`, ya que únicamente vamos a usar el cluster AKS con fines de desarrollo o demo.l.

Como mencione anteriormnete el código para la creación de este componente de infraestructura está definido dentro del archivo `recursos.tf`. Vamos a desglosar la configuración:

```
resource "azurerm_kubernetes_cluster" "aks_cluster" { name = "aks_cluster_kubernetes" location  
= azurerm_resource_group.acr_rg.location resource_group_name =  
azurerm_resource_group.acr_rg.name dns_prefix = "aks-dns-prefix"
```

El bloque de recurso "azurerm_kubernetes_cluster" define el recurso del clúster AKS. Se le asigna el nombre "aks_cluster".

El parámetro "name" especifica el nombre del clúster AKS como "aks_cluster_kubernetes".

El parámetro "location" especifica la ubicación (región de Azure) donde se creará el clúster AKS. Hace referencia a la ubicación de un grupo de recursos de Azure existente llamado "acr_rg" a través de la interpolación "azurerm_resource_group.acr_rg.location".

El parámetro "resource_group_name" especifica el nombre del grupo de recursos donde se creará el clúster AKS. Hace referencia al nombre de un grupo de recursos existente llamado "acr_rg" a través de la interpolación "azurerm_resource_group.acr_rg.name".

El parámetro "dns_prefix" especifica el prefijo DNS que se utilizará para el clúster AKS.

El bloque "identity" define la configuración de identidad para el clúster AKS.

El parámetro "type" especifica el tipo de identidad. En este caso, se establece en "SystemAssigned", lo que significa que Azure asignará una identidad administrada por el sistema al clúster AKS.

El bloque "default_node_pool" define la configuración del grupo de nodos predeterminado para el clúster AKS.

El parámetro "name" especifica el nombre del grupo de nodos predeterminado como "agentpool".

El parámetro "vm_size" especifica el tamaño de la máquina virtual para los nodos del grupo de nodos. Se establece en "Standard_D2_v2".

El parámetro "node_count" se espera que se proporcione como una variable, que determina el número de nodos en el grupo de nodos predeterminado.

Los detalles específicos, como la ubicación, el nombre del grupo de recursos y seel número de nodos proporcionan desde el archivo vars.tf para Terraform.

6.2 Descripción del frontend (AzureVote) y backend (Redis)

- AKS usará Azure Container Registry (ACR) para extraer imágenes (pull). Por tanto, AKS necesita autenticarse en ACR antes de realizar la acción de extracción o pull. Para resolver este tema se ha empleado Ansible, definiendo esta tarea en 03_playbook.yml .

El código de esta tarea es el siguiente:

Tarea 0 - Adjuntar ACR al AKS

- shell: Ejecuta comandos en la máquina local mediante la shell. En este caso, se utilizan comandos de la CLI de Azure (az) para autenticarse, actualizar el clúster AKS y adjuntar el ACR al clúster.
- az login: Autentica al usuario en Azure utilizando el nombre de usuario (AZURE_USERNAME), la contraseña (AZURE_PASSWORD) y el inquilino (AZURE_TENANT) proporcionados como variables de entorno.
- az aks update: Actualiza el clúster AKS especificado (kubernetes_cluster_name) en el grupo de recursos (resource_group_name) para adjuntar el ACR. El nombre de usuario del registro (registry_username) también se proporciona.
- Además, se establece become: true para ejecutar la tarea con privilegios de superusuario.

6.2 Descripción del frontend (AzureVote) y backend (Redis)

Frontend (AzureVote):

La referencia "mcr.microsoft.com/azuredocs/azure-vote-front:v1" es una imagen de contenedor alojada en el Registro de Contenedores de Microsoft (MCR).

Azure Vote es una aplicación de votación simple donde los usuarios pueden seleccionar opciones predefinidas y emitir sus votos.

Backend (Redis):

La referencia "mcr.microsoft.com/oss/bitnami/redis:6.0.8" es otra imagen de contenedor alojada en el Registro de Contenedores de Microsoft (MCR). Esta imagen proporciona Redis, una base de datos en memoria de código abierto y un sistema de almacenamiento en caché de clave-valor.

6.3 Customización de Imágenes, Push al Registry y Despliegue de Pods

Las acciones principales que se llevan a cabo en el 04_playbook.yml son :

Customización de Imágenes: Descarga de imágenes de contenedor desde repositorios públicos de Microsoft (Redis y Azure Vote Front) utilizando el comando podman pull.

Push al Registry: Etiquetado y subida de las imágenes customizadas al registro de contenedores de Azure utilizando el comando podman tag y podman push.

Despliegue de Pods: Creación de un namespace en el clúster de AKS y aplicación de un manifiesto YAML para desplegar los pods correspondientes en el clúster utilizando el módulo kubernetes.core.k8.

A continuación, se explica qué hace cada tarea:

Tarea 1 - Descargar imagen Redis desde el repositorio público de Microsoft:

Utiliza la biblioteca de Ansible containers.podman.podman_image para descargar la imagen mcr.microsoft.com/oss/bitnami/redis:6.0.8 desde el repositorio público de Microsoft.

Esta tarea requiere privilegios de root.

Tarea 2 - Descargar imagen Azure Vote Front desde el repositorio público de Microsoft:

Utiliza la biblioteca de Ansible containers.podman.podman_image para descargar la imagen mcr.microsoft.com/azuredocs/azure-vote-front:v1 desde el repositorio público de Microsoft.

Esta tarea requiere privilegios de root.

Tarea 3 - Etiquetar imagen Redis descargada:

Utiliza la biblioteca de Ansible containers.podman.podman_tag para etiquetar la imagen descargada de Redis con el nombre {{ registry_acr }}/redis:casopractico2.

{{ registry_acr }} es una variable que contiene el nombre del registro de contenedores.

Esta tarea requiere privilegios de root.

Tarea 4 - Etiquetar imagen Azure Vote Front descargada:

Utiliza el módulo de Ansible `containers.podman.podman_tag` para etiquetar la imagen descargada de Azure Vote Front con el nombre `{{ registry_acr }}/azurevote:casopractico2`.

`{{ registry_acr }}` es una variable que contiene el nombre del registro de contenedores.

Esta tarea requiere privilegios de root.

Tarea 5 - Crear namespace en el clúster de AKS:

Utiliza el módulo de Ansible `kubernetes.core.k8s` para crear un nuevo namespace llamado `azurevote-redis` en el clúster de AKS (Azure Kubernetes Service).

Esta tarea no requiere privilegios de root.

Tarea 8 - Iniciar sesión en el registro de contenedores:

Utiliza el módulo de Ansible `containers.podman.podman_login` para iniciar sesión en el registro de contenedores especificado por la variable `{{ registry_acr }}`.

Se proporcionan credenciales de inicio de sesión (username y password) para acceder al registro.

Esta tarea requiere privilegios de root.

Tarea 9 - Subir imagen Redis etiquetada al registro de contenedores de Azure:

Utiliza el módulo de Ansible `containers.podman.podman_image` para cargar la imagen etiquetada de Redis (`{{ registry_acr }}/redis:casopractico2`) en el registro de contenedores de Azure.

Esta tarea requiere privilegios de root.

Tarea 10 - Subir imagen Azure Vote Front etiquetada al registro de contenedores de Azure:

Utiliza el módulo de Ansible `containers.podman.podman_image` para cargar la imagen etiquetada de Azure Vote Front (`{{ registry_acr }}/azurevote:casopractico2`) en el registro de contenedores de Azure.

Esta tarea requiere privilegios de root.

Tarea 11 - Aplicar manifiesto YAML en el clúster de AKS:

Utiliza el módulo de Ansible `kubernetes.core.k8s` para aplicar el manifiesto YAML especificado por el archivo `manifest.yml` en el namespace `azurevote-redis` del clúster de AKS.

Esta tarea no requiere privilegios de root.

6.3 Consideraciones para el despliegue en AKS

Para que el despliegue de los pods se realice de forma correcta se requiere configurar en el mismo directorio los siguientes elementos:

`kubeconfig.yaml`

Configuración específica del clúster: Cada clúster de Kubernetes puede tener su propia configuración, como la dirección IP del servidor API, los certificados y las políticas de acceso. Al tener un `kubeconfig.yaml` separado para cada proyecto, se puede especificar la configuración única para ese clúster específico. También se pueden establecer permisos específicos para ese proyecto sin interferir con otros proyectos o clústeres. Por último Si se trabaja en múltiples proyectos o entornos de desarrollo con diferentes clústeres de Kubernetes, tener un `kubeconfig.yaml` separado para cada proyecto facilita la gestión y el cambio entre los diferentes contextos de clúster.

`vars.yml` :

- `kubernetes_cluster_name`: variable que contenga el nombre del clúster de Kubernetes.
- `resource_group_name`: variable que contenga el nombre del grupo de recursos de Azure.
- `registry_username`: variable que contenga el nombre de usuario del registro de Azure Container Registry (ACR) que se adjuntará al clúster de Kubernetes.

También se deberán definir :

- `AZURE_USERNAME`: variable de entorno contenga el nombre de usuario para iniciar sesión en Azure.
- `AZURE_PASSWORD`: variable de entorno contenga la contraseña para iniciar sesión en Azure.

- AZURE_TENANT: variable de entorno contenga el ID del inquilino (tenant) de Azure.

Manifest.yml

El archivo YAML describe la creación de un namespace llamado "azurevote-redis", un deployment que consta de un pod que ejecuta una aplicación llamada "azure-vote-back" con su correspondiente servicio, y otro deployment que consta de un pod que ejecuta una aplicación llamada "azure-vote-front" con su correspondiente servicio. Estos recursos son utilizados para implementar una aplicación de votación en Azure con una base de datos Redis.

Descripción de cada sección y lo que hace:

Namespace :

apiVersion: v1: Indica la versión de la API de Kubernetes que se está utilizando.

kind: Namespace: Especifica que se está creando un nuevo namespace (espacio de nombres).

metadata: Contiene metadatos asociados al namespace, como el nombre.

name: azurevote-redis: Es el nombre asignado al namespace.

Deployment :

apiVersion: apps/v1: Indica la versión de la API de aplicaciones de Kubernetes que se está utilizando.

kind: Deployment: Especifica que se está creando un nuevo deployment (implementación).

metadata: Contiene metadatos asociados al deployment, como el nombre y el namespace al que pertenece.

spec: Describe las características del deployment, como el número de réplicas, los selectores y la plantilla para los pods.

replicas: 1: Indica que se debe tener una réplica del pod.

selector: Define los labels (etiquetas) utilizados para seleccionar los pods a los que se aplicará este deployment.

template: Especifica la plantilla para crear los pods.

containers: Define los contenedores que se ejecutarán en los pods, incluyendo la imagen, los recursos y los puertos expuestos.

Service :

apiVersion: v1: Indica la versión de la API de Kubernetes que se está utilizando.

kind: Service: Especifica que se está creando un nuevo servicio.

metadata: Contiene metadatos asociados al servicio, como el nombre y el namespace al que pertenece.

spec: Describe las características del servicio, como los puertos expuestos y los selectores para enrutar el tráfico hacia los pods correspondientes.

7. Infraestructura como código con Terraform

7.1 Descripción de Terraform y su papel en el proyecto

Terraform es una herramienta poderosa que utilizo en mi proyecto para implementar la infraestructura como código. Su papel fundamental es permitirme definir y gestionar la infraestructura de manera programática y declarativa. Con Terraform, puedo describir mis recursos de infraestructura en archivos de configuración, lo que me permite mantener un control completo y reproducible de mi infraestructura en la nube.

En mi proyecto, utilizo Terraform para orquestar y provisionar los recursos necesarios en Microsoft Azure. Desde la creación de grupos de recursos y redes virtuales hasta la implementación de clústeres de Kubernetes, puedo definir y desplegar toda mi infraestructura de manera automatizada y coherente.

7.2 Estructura del código de infraestructura

La estructura de mi código de infraestructura en Terraform se organiza cuidadosamente para garantizar la claridad y la modularidad. Todos los archivos de configuración se encuentran dentro de un directorio principal llamado "terraform". Dentro de este directorio, tengo subdirectorios como ".terraform", "modules" y "providers", donde se ubican los archivos generados por Terraform, los módulos personalizados (si los hay) y los proveedores adicionales.

Los archivos principales se encuentran en el directorio raíz "terraform". Por ejemplo, el archivo "main.tf" contiene la configuración principal de los recursos de infraestructura que se van a crear, mientras que el archivo "outputs.tf" define las salidas que quiero mostrar después de aplicar la configuración de Terraform.

Además, tengo un archivo llamado "vars.tf" que contiene las variables utilizadas en el proyecto, como el nombre del grupo de recursos, la ubicación y la cantidad de nodos. Esto me permite personalizar y reutilizar mi configuración de Terraform de manera eficiente.

7.3 Organización y estructura del código de Terraform

La organización y estructura de mi código de Terraform se basa en las mejores prácticas recomendadas por la comunidad de Terraform. Sigo la convención de nombres y utilizo una estructura modular para mejorar la legibilidad y el mantenimiento del código.

Por ejemplo, agrupo los recursos relacionados en bloques lógicos dentro del archivo "recursos.tf". Defino el grupo de recursos, la red virtual, el registro de contenedores y la máquina virtual en secciones separadas, lo que facilita la comprensión y la gestión individual de cada recurso.

Además, separo las salidas deseadas en el archivo "outputs.tf", donde defino la información que quiero mostrar después de aplicar la configuración. En mi caso, muestro el nombre del grupo de recursos, la dirección IP pública de la máquina virtual y el nombre del clúster de Kubernetes.

En general, la estructura de mi código de Terraform sigue una jerarquía clara y coherente, con archivos y directorios bien organizados. Esto me permite mantener un código limpio y modular, facilitando la colaboración y el mantenimiento a largo plazo.

8. Configuración con Ansible

8.1 Descripción de Ansible y su papel en el proyecto

Ansible es una herramienta de automatización de TI ampliamente utilizada en proyectos de infraestructura. En este proyecto, Ansible desempeña un papel crucial al permitir la automatización de la configuración del sistema en diferentes entornos. Utilizando un enfoque declarativo, Ansible facilita la implementación y gestión de la configuración requerida en los nodos de destino.

En este proyecto, Ansible se utiliza para configurar y desplegar una infraestructura compuesta por una máquina virtual y un clúster de Azure Kubernetes Service (AKS). Ansible se encarga de instalar paquetes, generar certificados, configurar servicios web y desplegar contenedores en la máquina virtual, así como adjuntar un registro de contenedores (ACR) al clúster de AKS para el despliegue de un front-end y un back-end.

8.2 Organización y estructura del código de Ansible

El directorio raíz del proyecto de Ansible está estructurado de la siguiente manera:

```
/ansible /templates .htaccess Containerfile httpd.conf index.html 00_playbook.yml  
01_playbook.yml 02_playbook.yml 03_playbook.yml 04_playbook.yml deploy.sh host.txt  
kubeconfig.yaml manifest.yaml vars.yaml
```

Cada archivo y directorio tiene un propósito específico:

El directorio "templates" contiene plantillas de archivos utilizadas en la configuración y generación de archivos de configuración.

Los "playbooks" numéricos del "00_playbook.yml" al "04_playbook.yml" son responsables de diferentes configuraciones y tareas en el proyecto.

El archivo "deploy.sh" es un script utilizado para desplegar y ejecutar los "playbooks" de Ansible.

El archivo "host.txt" especifica la dirección IP y las credenciales de autenticación SSH para la máquina virtual en la que se realizarán las configuraciones.

El archivo "kubeconfig.yaml" es un archivo de configuración de Kubernetes utilizado para la interacción con el clúster de AKS.

El archivo "manifest.yml" contiene un manifiesto YAML que describe los recursos que se desplegarán en el clúster de AKS.

El archivo "vars.yml" define las variables utilizadas en los "playbooks" de Ansible, como "registry_acr", "registry_username", "registry_password", "kubernetes_cluster_name", "resource_group_name", "azure_client_id", "azure_tenant", "azure_name", "azure_pwd" y "ansible_become_pass".

8.3 Automatización de la configuración del sistema

Cada "playbook" de Ansible se encarga de tareas específicas para configurar el sistema en los nodos de destino. A continuación, se proporciona una descripción general de cada "playbook":

"00_playbook.yml": Este "playbook" configura una máquina virtual específica ("podman_vm"). Instala herramientas como "podman", "skopeo", "httpd", "httpd-tools" y "openssl", además de crear directorios y generar certificados necesarios para el servicio web.

"01_playbook.yml": Este "playbook" se encarga de la configuración del servidor Apache en la máquina virtual. Genera archivos como "index.html", "httpd.conf", ".htaccess" y "Containerfile" a partir de plantillas ubicadas en el directorio "templates".

"02_playbook.yml": Este "playbook" se enfoca en la creación de imágenes de contenedores y su despliegue. Genera una imagen del contenedor a partir de un "Containerfile" y la sube a un registro de contenedores utilizando credenciales específicas. Luego, crea y administra el contenedor en la máquina virtual.

"03_playbook.yml": Este "playbook" adjunta un registro de contenedores (ACR) al clúster de AKS. Utiliza credenciales y detalles de configuración específicos para autenticarse en Azure y asociar el ACR al clúster de AKS.

"04_playbook.yml": Este "playbook" se centra en la personalización de imágenes de contenedores, la carga de las imágenes personalizadas en el registro de contenedores y el despliegue de pods en el clúster de AKS. Descarga imágenes públicas de Microsoft, etiqueta las imágenes y las carga en el registro de contenedores. Luego, aplica un manifiesto YAML para desplegar los recursos en el

clúster de AKS.

9. Proceso de despliegue automatizado

El proceso de despliegue automatizado es un conjunto de pasos y acciones secuenciales que se ejecutan de forma automatizada para implementar y configurar una infraestructura o aplicación de manera consistente y confiable. Este proceso es fundamental para agilizar y simplificar el despliegue, minimizando los errores humanos y garantizando la coherencia en todos los entornos.

9.1 Scripts de despliegue automatizado

En el contexto de este proyecto, se han utilizado scripts de despliegue automatizado para llevar a cabo las tareas necesarias. Estos scripts contienen una serie de comandos y acciones que se ejecutan de forma secuencial para lograr la configuración y el despliegue esperados. A continuación, se muestra el script de despliegue automatizado:

```
#!/bin/bash
```

```
# Eliminar una clave de host específica del archivo known_hosts
```

```
ssh-keygen -f "/home/server_admin/.ssh/known_hosts" -R "20.254.107.64"
```

```
# Añade a known_hosts para evitar la verificación del fingerprint al conectarte a ese host
```

```
ssh-keyscan -H 20.254.107.64 >> ~/.ssh/known_hosts
```

```
# Asegúrate de que ansible/kubeconfig.yaml es el correcto
```

```
yes | az aks get-credentials --resource-group cp2_resource_group --name aks_cluster_kubernetes --  
file /mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible/kubeconfig.yaml --overwrite-  
existing
```

```
# Asegúrate de que ~/.kube/config es el correcto
```

```
yes | az aks get-credentials --resource-group cp2_resource_group --name aks_cluster_kubernetes --
```

```
file ~/.kube/config --overwrite-existing
```

```
# Verificar y ajustar los permisos del archivo kubeconfig
```

```
chmod 600 ~/.kube/config
```

```
# Lista de playbooks a ejecutar para la VM con Podman
```

```
ansible-playbook -i hosts.txt 00_playbook.yml --extra-vars "@vars.yml"
```

```
ansible-playbook -i hosts.txt 01_playbook.yml --extra-vars "@vars.yml"
```

```
ansible-playbook -i hosts.txt 02_playbook.yml --extra-vars "@vars.yml"
```

```
# Lista de playbooks a ejecutar para la AzureVote con Redis
```

```
ansible-playbook -i hosts.txt 03_playbook.yml --extra-vars "@vars.yml"
```

```
ansible-playbook -i hosts.txt 04_playbook.yml --extra-vars "@vars.yml"
```

En este script, se llevan a cabo una serie de acciones automatizadas. Por ejemplo, se eliminan y añaden claves de host en el archivo "known_hosts" para establecer la comunicación segura con la máquina virtual. Se asegura que los archivos de configuración de Ansible y Kubernetes, como "kubeconfig.yml" y "~/.kube/config", estén correctamente configurados y tengan los permisos adecuados. Luego, se ejecutan los "playbooks" de Ansible, especificados en la lista, para configurar la máquina virtual con Podman y desplegar la aplicación AzureVote con Redis en el clúster de AKS.

Este script de despliegue automatizado permite ejecutar de manera rápida y eficiente las tareas necesarias para implementar y configurar el entorno deseado. Al automatizar estos pasos, se garantiza la coherencia y se reducen los errores manuales, lo que facilita el proceso de despliegue y mejora la eficiencia del proyecto.

10. Pruebas y validación

10.1 Estrategia de pruebas utilizada

Para comprobar que mi despliegue de servidor contenerizado y mi clúster de AKS funcionen correctamente según los playbooks que hemos analizado he realiza las siguientes pruebas:

Revisión general de los recursos creados:

<input type="checkbox"/> Nombre ↑↓	Tipo ↑↓	Grupo de recursos ↑↓	Ubicación ↑↓
<input type="checkbox"/> podmanVM_OsDisk_1_fdcf6d4a84b94033bc22f67eb7d32a67	Disco	CP2_RESOURCE_GROUP	UK South
<input type="checkbox"/> podmanVM	Máquina virtual	cp2_resource_group	UK South
<input type="checkbox"/> maseiraacr	Registro de contenedor	cp2_resource_group	UK South
<input type="checkbox"/> aks_cluster_kubernetes	Servicio de Kubernetes	cp2_resource_group	UK South
<input type="checkbox"/> podmanVM_nic	Interfaz de red	cp2_resource_group	UK South
<input type="checkbox"/> podmanVM_public_ip	Dirección IP pública	cp2_resource_group	UK South
<input type="checkbox"/> vnet	Red virtual	cp2_resource_group	UK South
<input type="checkbox"/> aks-agentpool-20772406-vmss	Conjunto de escalas de máquina virtual	MC_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> aks_cluster_kubernetes-agentpool	Identidad administrada	MC_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> c38f38d1-1c43-4170-8b1f-81ef05c70517	Dirección IP pública	MC_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> aks-agentpool-28408381-routetable	Tabla de rutas	MC_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> aks-vnet-28408381	Red virtual	MC_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> kubernetes	Equilibrador de carga	mc_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> aks-agentpool-28408381-nsg	Grupo de seguridad de red	mc_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> kubernetes-a9e4d0f415e9a4fe3b27fc40d3a18eb4	Dirección IP pública	mc_cp2_resource_group_aks_cluster_ku...	UK South
<input type="checkbox"/> NetworkWatcher_uksouth	Network Watcher	NetworkWatcherRG	UK South

Verificación de la máquina virtual:

- Comprobar que la máquina virtual se haya creado correctamente y esté en un estado activo.

Conectar ▶ Iniciar ↺ Reiniciar □ Detener 📷 Captura 🗑️ Eliminar ↻ Actualizar 📱 Abrir en dispositivos móviles ⋮

^ Información esencial [Vista JSON](#)

Grupo de recursos (mover) cp2_resource_group	Sistema operativo Linux (centos 8.5.2111)
Estado En ejecución	Tamaño Standard DS1 v2 (1 vcpu, 3.5 GiB de memoria)
Ubicación UK South	Dirección IP pública 20.254.107.64
Suscripción (mover) Azure for Students	Red virtual/subred vnet/subnet
Id. de suscripción c8b29887-4024-4277-8a83-a951bff463b3	Nombre DNS Sin configurar
	Estado de mantenimiento -

Etiquetas ([editar](#))
[Haga clic aquí para agregar etiquetas.](#)

Propiedades Supervisión Funcionalidades (7) Recomendaciones Tutoriales

Máquina virtual

Nombre del equipo	podmanVM
Sistema operativo	Linux (centos 8.5.2111)
Editor de imagen	cognosys
Oferta de imagen	centos-8-stream-free
Plan de imagen	centos-8-stream-free
Generación de VM	V1
Arquitectura de VM	x64
Estado del agente	Ready
Versión del agente	2.9.1.1
Grupo host	Ninguno
Host	-
Grupo con ubicación por proximidad	-
Estado de ubicación	N/D
Grupo de reserva de capacidad	-
Tipo de controladora	-

Redes

Dirección IP pública	20.254.107.64 (Interfaz podmanVM_nic de red)
Dirección IP pública (IPv6)	-
Dirección IP privada	10.0.0.4
Dirección IP privada (IPv6)	-
Red virtual/subred	vnet/subnet
Nombre DNS	Configurar

Tamaño

Tamaño	Standard DS1 v2
vCPU	1
RAM	3.5 GiB

Disco

Disco del SO	podmanVM_OsDisk_1_fd3cf6d4a84b94033bc22f67eb7d32a67
--------------	---

- Acceder a la dirección IP pública de la máquina virtual y verificar que el servicio web esté en funcionamiento.

← → ↻ ⚠ No es seguro | <https://20.254.107.64:8080> [🔗](#) [★](#)

Bienvenido al servidor web

Este servidor web ha sido configurado con Podman y Ansible.

A continuación se muestran algunas características del servidor:

- Se ha instalado y configurado el servidor web Apache HTTP Server (httpd).
- El servidor web utiliza un certificado autofirmado para conexiones seguras (HTTPS).
- Se ha configurado la autenticación básica para proteger el acceso al servidor web.
- Se ha generado una imagen del contenedor con todas las configuraciones que corre sobre Podman y se ha subido al registro de contenedores de Azure.
- El contenedor del servicio web se gestiona a través de systemd para automatizar su puesta en marcha en caso de fallo o reinicio.


Este servidor es parte del casopractico2 que estoy desarrollando para Unir.

- Acceder a la dirección IP pública de la máquina virtual y verificar que el estado del contenedor:

```
[adminuser@podmanVM ~]$ sudo podman ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
a5ea04925560   maseiraacr.azurecr.io/webserver:casopractico2  httpd-foreground        8 minutes ago  Up 8 minutes ago  0.0.0.0:8080->443/tcp
[adminuser@podmanVM ~]$
```

- Acceder al contenedor y modificar el index.html que sirve la máquina

```
root@a5ea04925560:/usr/local/apache2# sed -i 's/casopractico2/casopractico3/g' /usr/local/apache2/htdocs/index.html
root@a5ea04925560:/usr/local/apache2#
```



Bienvenido al servidor web

Este servidor web ha sido configurado con Podman y Ansible.

A continuacion se muestran algunas características del servidor:

- Se ha instalado y configurado el servidor web Apache HTTP Server (httpd).
- El servidor web utiliza un certificado autofirmado para conexiones seguras (HTTPS).
- Se ha configurado la autentificacion basica para proteger el acceso al servidor web.
- Se ha generado una imagen del contenedor con todas las configuraciones que corre sobre Podman y se ha subido al registro de contenedores de Azure.
- El contenedor del servicio web se gestiona a traves de systemd para automatizar su puesta en marcha en caso de fallo o reinicio.

Este servidor es parte del casopractico3 que estoy desarrollando para Unir.

Comprobación del clúster de AKS:

- Verificar que el clúster de AKS se haya creado correctamente y esté en un estado activo.

aks_cluster_kubernetes
Servicio de Kubernetes

+ Crear | Conectar | Inicio | Detener | Eliminar | Actualizar | Abrir en dispositivos móviles | Enviar comentarios

Información esencial

Grupo de recursos : [cp2_resource_group](#)
 Estado : Correcto (En ejecución)
 Ubicación : UK South
 Suscripción : [Azure for Students](#)
 Id. de suscripción : c8b29887-4024-4277-8a83-a951bf463b3
 Etiquetas (edita) : [Haga clic aquí para agregar etiquetas.](#)

Versión de Kubernetes : [1.25.6](#)
 Dirección del servidor de... : [aks-dns-prefix-v9jopo8h.hcp.uksouth.azmk8s.io](#)
 Tipo de red (complement... : [Kubenet](#)
 Grupos de nodos : [1 grupo de nodos](#)

Comenzar | **Propiedades** | Supervisión | Funcionalidades (4) | Recomendaciones | Tutoriales

Servicios de Kubernetes

Tipo de cifrado : Cifrado en reposo con una clave administrada por la plataforma
 Grupos de nodos virtuales : No habilitado

Grupos de nodos

Grupos de nodos : 1 grupo de nodos
 Versiones de Kubernetes : 1.25.6
 Tamaños de nodo : Standard_D2_v2

Configuración

Versión de Kubernetes : 1.25.6
 Tipo de actualización automática : Deshabilitado
 Autenticación y autorización : Cuentas locales con RBAC de Kubernetes
 Cuentas locales : Habilitado

Extensiones + aplicaciones

No hay extensiones instaladas

Redes

Dirección del servidor de API : [aks-dns-prefix-v9jopo8h.hcp.uksouth.azmk8s.io](#)
 Tipo de red (complemento) : Kubenet
 CIDR del pod : 10.244.0.0/16
 CIDR de servicio : 10.0.0.0/16
 Dirección IP del servicio DNS : 10.0.0.10
 CIDR del puente de Docker : -
 Directiva de red : Ninguno
 Equilibrador de carga : Standard
 Enrutamiento de solicitudes HTTP : No habilitado
 Clúster privado : No habilitado
 Intervalos IP autorizados : No habilitado
 Controlador de entrada de Application Gateway : No habilitado

Integraciones

Información de contenedores : No habilitado
 Id. del recurso del área de trabajo : -

- Revisar que todas las partes del clúster estén en funcionamiento y en el estado esperado.

+ Crear | Eliminar | Actualizar | Mostrar etiquetas | Enviar comentarios

Implementaciones | Pods | Conjuntos de réplicas | Objetos StatefulSet | Conjuntos de demonios | Trabajos | Trabajos cron

Filtrar por nombre de implementación :
 Filtrar por espacio de nombres : [Agregar filtro de etiqueta](#)

<input type="checkbox"/>	Nombre	Espacio de nombres	Listo	Actualizadas	Disponible	Antigüedad ↓
<input type="checkbox"/>	coredns	kube-system	✓ 2/2	2	2	2 horas
<input type="checkbox"/>	coredns-autoscaler	kube-system	✓ 1/1	1	1	2 horas
<input type="checkbox"/>	kubernetes-agent	kube-system	✓ 2/2	2	2	2 horas
<input type="checkbox"/>	metrics-server	kube-system	✓ 2/2	2	2	2 horas
<input type="checkbox"/>	azure-vote-back	azurevote-redis	✓ 1/1	1	1	1 hora
<input type="checkbox"/>	azure-vote-front	azurevote-redis	✓ 1/1	1	1	1 hora

aks_cluster_kubernetes | Cargas de trabajo

Implementaciones Pods Conjuntos de réplicas Objetos StatefulSet Conjuntos de demonios Trabajos Trabajos cron

Filtrar por nombre de pod Estado Filtrar por espacio de nombres

Nombre	Espacio de nombres	Listo	Estado	Reiniciar recue...	Antigüedad	IP del pod
coredns-autoscaler-69b7556b86-h94vz	kube-system	1/1	Running	0	2 horas	10.244.0.3
coredns-fb6b9d95f-zlcz	kube-system	1/1	Running	0	2 horas	10.244.0.5
azure-ip-masq-agent-6r6tt	kube-system	1/1	Running	0	2 horas	10.224.0.4
cloud-node-manager-bt4x8	kube-system	1/1	Running	0	2 horas	10.224.0.4
csi-azuredisk-node-8txmn	kube-system	3/3	Running	0	2 horas	10.224.0.4
csi-azurefile-node-jlvhc	kube-system	3/3	Running	0	2 horas	10.224.0.4
kube-proxy-pcw9z	kube-system	1/1	Running	0	2 horas	10.224.0.4
coredns-fb6b9d95f-wwv4z	kube-system	1/1	Running	0	2 horas	10.244.0.8
metrics-server-67db6db9b5-b8p5t	kube-system	2/2	Running	0	2 horas	10.244.0.9
metrics-server-67db6db9b5-cfg6w	kube-system	2/2	Running	0	2 horas	10.244.0.10
konnectivity-agent-86666bc88c-dqhtv	kube-system	1/1	Running	0	1 hora	10.244.0.13
konnectivity-agent-86666bc88c-v8tzh	kube-system	1/1	Running	0	1 hora	10.244.0.14
azure-vote-back-977f4c987-znzdt	azurevote-redis	1/1	Running	0	1 hora	10.244.0.11
azure-vote-front-7f5647fd4-n96z6	azurevote-redis	1/1	Running	0	1 hora	10.244.0.12

aks_cluster_kubernetes | Cargas de trabajo

Implementaciones Pods Conjuntos de réplicas Objetos StatefulSet Conjuntos de demonios Trabajos Trabajos cron

Filtrar por nombre de conjunto de réplicas Filtrar por espacio de nombres

Nombre	Espacio de nombres	Listo	Actual	Antigüedad	Imágenes
coredns-autoscaler-69b7556b86	kube-system	1/1	1	2 horas	mcr.microsoft.com/oss/kubernet...
coredns-fb6b9d95f	kube-system	2/2	2	2 horas	mcr.microsoft.com/oss/kubernet...
konnectivity-agent-7cd8b9cfc6	kube-system	0/0	0	2 horas	mcr.microsoft.com/oss/kubernet...
metrics-server-6589d797c	kube-system	0/0	0	2 horas	mcr.microsoft.com/oss/kubernet...
metrics-server-67db6db9b5	kube-system	2/2	2	2 horas	mcr.microsoft.com/oss/kubernet...
konnectivity-agent-86666bc88c	kube-system	2/2	2	1 hora	mcr.microsoft.com/oss/kubernet...
azure-vote-back-977f4c987	azurevote-redis	1/1	1	1 hora	maseiraacr.azurecr.io/redisasop...
azure-vote-front-7f5647fd4	azurevote-redis	1/1	1	1 hora	maseiraacr.azurecr.io/azurevote...

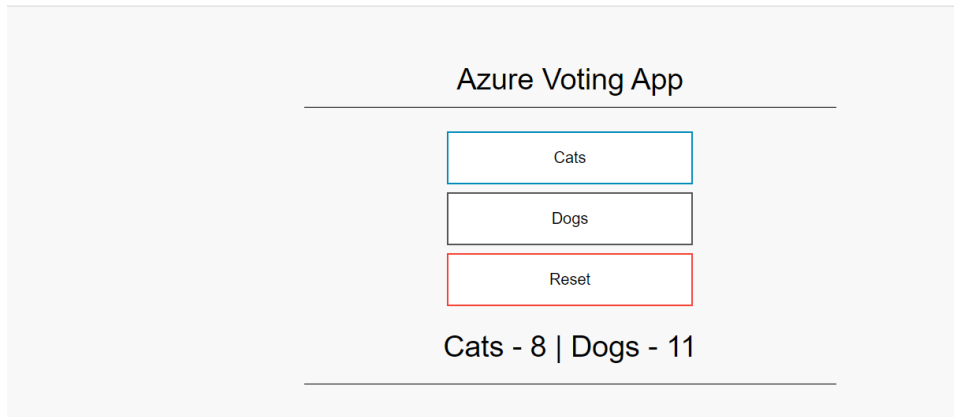
- Revisar servicios expuestos por los pods y verifica que respondan correctamente.

aks_cluster_kubernetes | Servicios y entradas

Servicios Entradas

Filtrar por nombre de servicio Filtrar por espacio de nombres

Nombre	Espacio de nombres	Estado	Tipo	IP del clúster	IP externa	Puertos	Antigüedad
kubernetes	default	Correcto	ClusterIP	10.0.0.1		443/TCP	2 horas
kube-dns	kube-system	Correcto	ClusterIP	10.0.0.10		53/UDP,53/TCP	2 horas
metrics-server	kube-system	Correcto	ClusterIP	10.0.75.177		443/TCP	2 horas
azure-vote-back	azurevote-redis	Correcto	ClusterIP	10.0.234.118		6379/TCP	2 horas
azure-vote-front	azurevote-redis	Correcto	LoadBalancer	10.0.182.207	20.68.186.216	8032710/TCP	2 horas



Pruebas de extremo a extremo:

- Verificar que los datos se almacenen y recuperen correctamente de la base de datos o almacenamiento utilizado por la aplicación.

```
server_admin@OAlvarez-Desl x + v
azure-vote-back-977f4c987-znzd 1/1 Running 0 115m
azure-vote-front-7f5647fd4-n96z6 1/1 Running 0 115m
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible$ kubectl logs -n azurevote-r
edis azure-vote-back-977f4c987-znzd
redis 09:58:00.62
redis 09:58:00.62 Welcome to the Bitnami redis container
redis 09:58:00.64 Subscribe to project updates by watching https://github.com/bitnami/bitnami-docker-redis
redis 09:58:00.64 Submit issues and feature requests at https://github.com/bitnami/bitnami-docker-redis/issues
redis 09:58:00.64
redis 09:58:00.64 INFO ==> ** Starting Redis setup **
redis 09:58:00.66 WARN ==> You set the environment variable ALLOW_EMPTY_PASSWORD=yes. For safety reasons, do not use
this flag in a production environment.
redis 09:58:00.66 INFO ==> Initializing Redis
redis 09:58:00.75 INFO ==> Setting Redis config file
redis 09:58:00.85 INFO ==> ** Redis setup finished! **

redis 09:58:00.86 INFO ==> ** Starting Redis **
1:C 17 Jul 2023 09:58:00.952 # o000o000o000o Redis is starting o000o000o000o
1:C 17 Jul 2023 09:58:00.952 # Redis version=6.0.8, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 17 Jul 2023 09:58:00.952 # Configuration loaded

Redis 6.0.8 (00000000/0) 64 bit
Running in standalone mode
Port: 6379
PID: 1
http://redis.io

1:M 17 Jul 2023 09:58:00.953 # Server initialized
1:M 17 Jul 2023 09:58:00.953 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will
create latency and memory usage issues with Redis. To fix this issue run the command 'echo madvise > /sys/kernel/
mm/transparent_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting after a re
boot. Redis must be restarted after THP is disabled (set to 'madvise' or 'never').
1:M 17 Jul 2023 09:58:00.954 * Ready to accept connections
1:M 17 Jul 2023 10:13:01.087 * 1 changes in 900 seconds. Saving...
1:M 17 Jul 2023 10:13:01.087 * Background saving started by pid 39
39:C 17 Jul 2023 10:13:01.107 * DB saved on disk
39:C 17 Jul 2023 10:13:01.107 * RDB: 0 MB of memory used by copy-on-write
1:M 17 Jul 2023 10:13:01.188 * Background saving terminated with success
1:M 17 Jul 2023 11:32:53.438 * 1 changes in 900 seconds. Saving...
1:M 17 Jul 2023 11:32:53.438 * Background saving started by pid 40
40:C 17 Jul 2023 11:32:53.447 * DB saved on disk
40:C 17 Jul 2023 11:32:53.447 * RDB: 0 MB of memory used by copy-on-write
1:M 17 Jul 2023 11:32:53.538 * Background saving terminated with success
1:M 17 Jul 2023 11:37:54.051 * 10 changes in 300 seconds. Saving...
1:M 17 Jul 2023 11:37:54.052 * Background saving started by pid 41
41:C 17 Jul 2023 11:37:54.062 * DB saved on disk
41:C 17 Jul 2023 11:37:54.062 * RDB: 0 MB of memory used by copy-on-write
1:M 17 Jul 2023 11:37:54.152 * Background saving terminated with success
```

```
Azure Voting App
Cats
Dogs
Reset
Cats - 8 | Dogs - 11

server_admin@OAlvarez-Desl x + v
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible$ kubectl exec -n azurevote-redis -it
azure-vote-back-977f4c987-znzd -- redis-cli KEYS "*"
1) "Cats"
2) "Dogs"
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible$ kubectl exec -n azurevote-redis -it
azure-vote-back-977f4c987-znzd -- redis-cli GET Cats
8
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible$ kubectl exec -n azurevote-redis -it
azure-vote-back-977f4c987-znzd -- redis-cli GET Dogs
11
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2/ansible$
```

Pruebas de disponibilidad:

Se han realizado pruebas sobre la capacidad de recuperación del sistema al realizar acciones como la caída del podmanVM o fallas en los contenedores, verificando que los sistemas se recuperan automáticamente sin interrupciones significativas en el servicio.

10.2 Monitorización y registro

En el proyecto, se reconoce la importancia de implementar supervisión y registro como buenas prácticas para garantizar el rendimiento y la disponibilidad de los sistemas desplegados. Aunque no se ha llevado a cabo ninguna acción específica en este sentido puesto que no es un criterio evaluable, se pueden considerar diversas opciones de integración para lograr la monitorización y registro adecuados en las siguientes áreas:

Supervisión de PodmanVM

Para supervisar la máquina virtual donde se ejecuta el entorno de Podman, se podría integrar una solución de monitorización de infraestructura, como Prometheus, Grafana o Azure Monitor. Estas herramientas permiten recopilar métricas clave del sistema, como el uso de recursos, la carga del sistema y la latencia de red.

Supervisión de AKS

Para supervisar el clúster de Kubernetes administrado por Azure Kubernetes Service (AKS), se puede utilizar Azure Monitor, que proporciona métricas y registros detallados sobre el clúster y sus recursos. Esto permite realizar un seguimiento del rendimiento, la capacidad y la disponibilidad del clúster de AKS.

Registro centralizado de logs y eventos

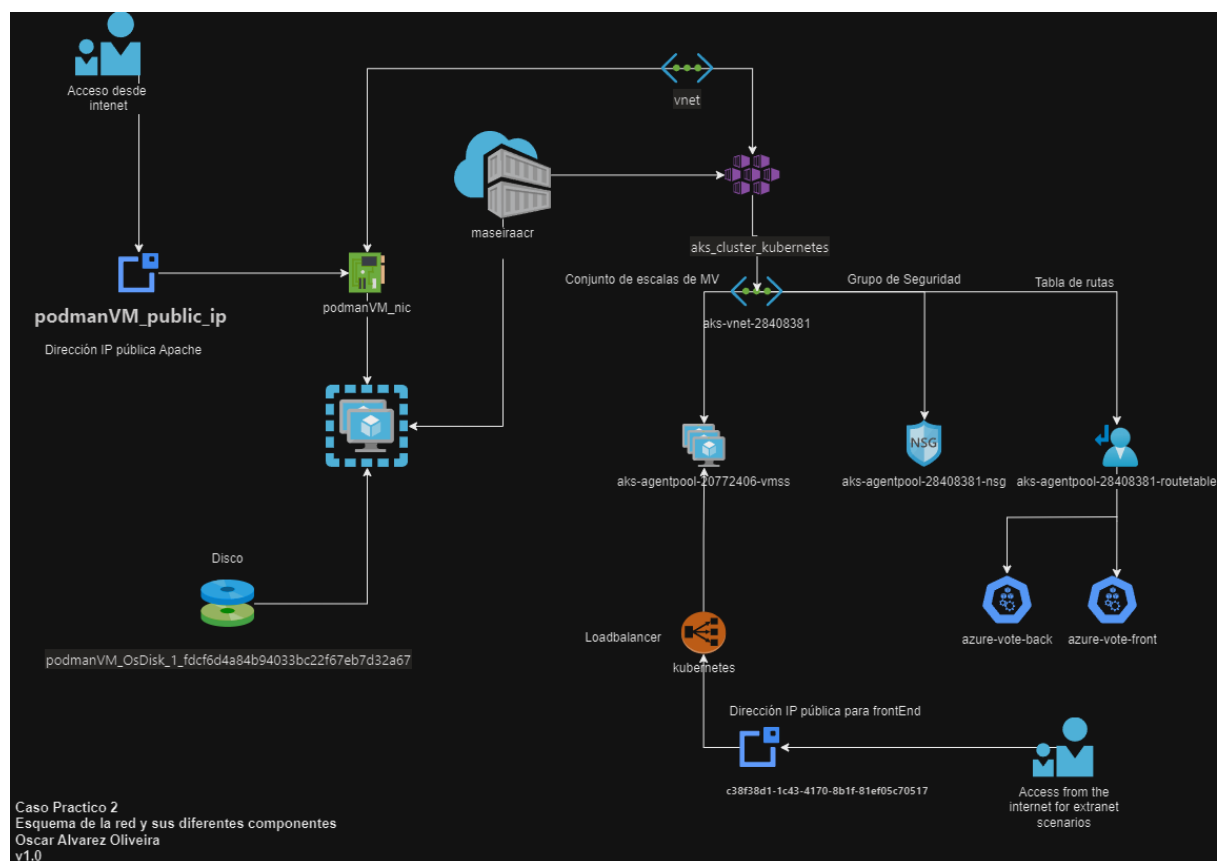
Para recopilar y analizar los registros generados por los componentes desplegados, se puede integrar una plataforma de registros centralizada, como ELK (Elasticsearch, Logstash, Kibana) o Azure Monitor Logs. Estas soluciones facilitan la agregación, búsqueda y análisis de logs en tiempo real, lo que simplifica la detección de problemas y las investigaciones.

Alertas y notificaciones

Además de la supervisión y el registro, se puede implementar un sistema de alertas y notificaciones para recibir notificaciones automáticas cuando se detecten eventos o condiciones anormales. Esto se puede lograr utilizando herramientas como Prometheus Alertmanager, Azure Monitor Alerts o servicios de notificaciones como Slack o Microsoft Teams.

11. Anexos

11.1 Diagramas de arquitectura



11.2 Capturas de pantalla

Instalar terraform 1.5.2 en HOST


```
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ sudo snap install terraform --classic
terraform 1.5.2 from Jon Seager (jnsgruk) installed
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ terraform version
Terraform v1.5.2
on linux_amd64
```

Instalar azure CLI en HOST

```
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ az --version
azure-cli                2.49.0

core                     2.49.0
telemetry                1.0.8

Dependencies:
msal                     1.20.0
azure-mgmt-resource      22.0.0

Python location '/opt/az/bin/python3'
Extensions directory '/home/server_admin/.azure/cliextensions'

Python (Linux) 3.10.10 (main, May 19 2023, 08:20:31) [GCC 11.3.0]

Legal docs and information: aka.ms/AzureCliLegal

Your CLI is up-to-date.
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ |
```

Instalar Ansibe en HOST

```
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ az --version
azure-cli                2.49.0

core                     2.49.0
telemetry                1.0.8

Dependencies:
msal                     1.20.0
azure-mgmt-resource      22.0.0

Python location '/opt/az/bin/python3'
Extensions directory '/home/server_admin/.azure/cliextensions'

Python (Linux) 3.10.10 (main, May 19 2023, 08:20:31) [GCC 11.3.0]

Legal docs and information: aka.ms/AzureCliLegal

Your CLI is up-to-date.
server_admin@OAlvarez-Desktop:/mnt/d/Minhasse/Documents/Proyectos/casopractico2$ |
```

Otras capturas de pantalla relevantes, pero de difícil visualización serán incluidas como archivos png para su consulta.

- terraform apply
- Terraform plan
- Run del ssh

12. Lecciones Aprendidas

12.1 Conocimientos técnicos adquiridos

Dominio de Terraform: Durante el proyecto, he aprendido a utilizar Terraform para describir y aprovisionar la infraestructura en Azure de forma automatizada. He adquirido conocimientos sobre la sintaxis de Terraform, la gestión de recursos y la configuración de proveedores.

Uso de Ansible: Mediante el uso de Ansible, he aprendido a automatizar la configuración y el despliegue de la máquina virtual y los contenedores en Azure. He desarrollado habilidades en la definición de playbooks, roles y tareas de Ansible, así como en la gestión de la configuración.

Implementación de contenedores con Podman: He adquirido conocimientos sobre la utilización de Podman como motor de contenedores para desplegar y administrar aplicaciones contenerizadas en la máquina virtual. He aprendido a construir y gestionar imágenes de contenedor, así como a orquestar los contenedores en la máquina virtual.

12.2 Comprensión teórica alcanzada

Arquitectura de Azure: A lo largo del proyecto, he profundizado en la comprensión de la arquitectura de Azure, incluyendo conceptos como los servicios de Azure, los grupos de recursos y la conectividad entre ellos. He adquirido un conocimiento sólido sobre los componentes necesarios para desplegar y gestionar una infraestructura en la nube de Azure.

Despliegue de servicios en Azure Kubernetes Service (AKS): He estudiado y comprendido los conceptos fundamentales de AKS, como los clústeres de Kubernetes, los nodos, los pods y los servicios. He aprendido a integrar un clúster de AKS con Azure Container Registry (ACR) para almacenar y desplegar imágenes de contenedor de manera eficiente.

12.3 Experiencia práctica obtenida

Implementación de un servidor web contenerizado: A través del proyecto, he adquirido experiencia en el despliegue de un servidor web contenerizado en una máquina virtual de Azure utilizando Podman. He comprendido los desafíos relacionados con la configuración y el mantenimiento de los contenedores, así como la interacción con otros servicios de Azure.

Configuración de un clúster de AKS y ACR: He adquirido habilidades prácticas para la configuración y administración de un clúster de AKS y su integración con ACR. He aprendido a desplegar y escalar los servicios en el clúster, así como a gestionar las imágenes de contenedor almacenadas en el registro de ACR.

Investigación de mejores prácticas: Durante el proyecto, he desarrollado habilidades de investigación para identificar las mejores prácticas y las recomendaciones de seguridad relacionadas con la implementación de una infraestructura en Azure. He aprendido a buscar y aplicar soluciones a los desafíos técnicos encontrados, así como a adaptar la configuración según las necesidades del proyecto.

12.4 Desafíos

Integración y orquestación de servicios: Uno de los desafíos principales fue integrar y orquestar correctamente los servicios de Azure, como la máquina virtual, AKS, ACR y Redis. Aprendí la importancia de comprender las dependencias y las configuraciones necesarias para que los servicios funcionen de manera conjunta.

12.5 Gestión de la configuración y errores

Durante el proyecto, me encontré con errores y dificultades en la gestión de la configuración de los recursos en Azure. Aprendí a depurar y solucionar problemas, así como a tener un enfoque proactivo en la gestión de la configuración para evitar problemas futuros.

13.Licencia

Este proyecto se encuentra bajo la licencia MIT. Consulte el archivo LICENSE para obtener más información sobre las restricciones y el uso permitido por esta licencia.

13.1 La elección de la licencia MIT

En primer lugar, al optar por la licencia MIT, estoy promoviendo la apertura y la colaboración en el ámbito del software. Esta licencia de código abierto permite a otros utilizar, modificar y distribuir el software que he desarrollado como parte de mi proyecto. Reconozco la importancia de compartir conocimientos y recursos, y al elegir esta licencia, estoy fomentando un ambiente propicio para la colaboración.

Además, es crucial destacar el papel de la comunidad de código abierto en el éxito de mi proyecto. Sin la invaluable contribución de la comunidad de desarrolladores y colaboradores de software libre, mi trabajo no habría sido posible. A través de la licencia MIT, expreso mi profundo agradecimiento a esta comunidad, permitiendo que otros se beneficien de mi trabajo y, a su vez, contribuyan con mejoras o utilicen mi proyecto como base para sus propios desarrollos.

Asimismo, quiero resaltar el apoyo invaluable del claustro de profesores de mi experto en el desarrollo de mi proyecto. Sus conocimientos y orientación han sido fundamentales para alcanzar los resultados obtenidos. Al emplear la licencia MIT, también brindo reconocimiento y agradecimiento a mi claustro de profesores, permitiéndoles utilizar y compartir mi proyecto en su propio trabajo o investigación.