Administración de sistemas

Vagrant + Kubernetes + Traefik

EKAIN AGUIRREZABAL DIAZ
Curso 2017/20189
Facultad de Informática, UPV/EHU
Vitoria-Gasteiz

INDICE

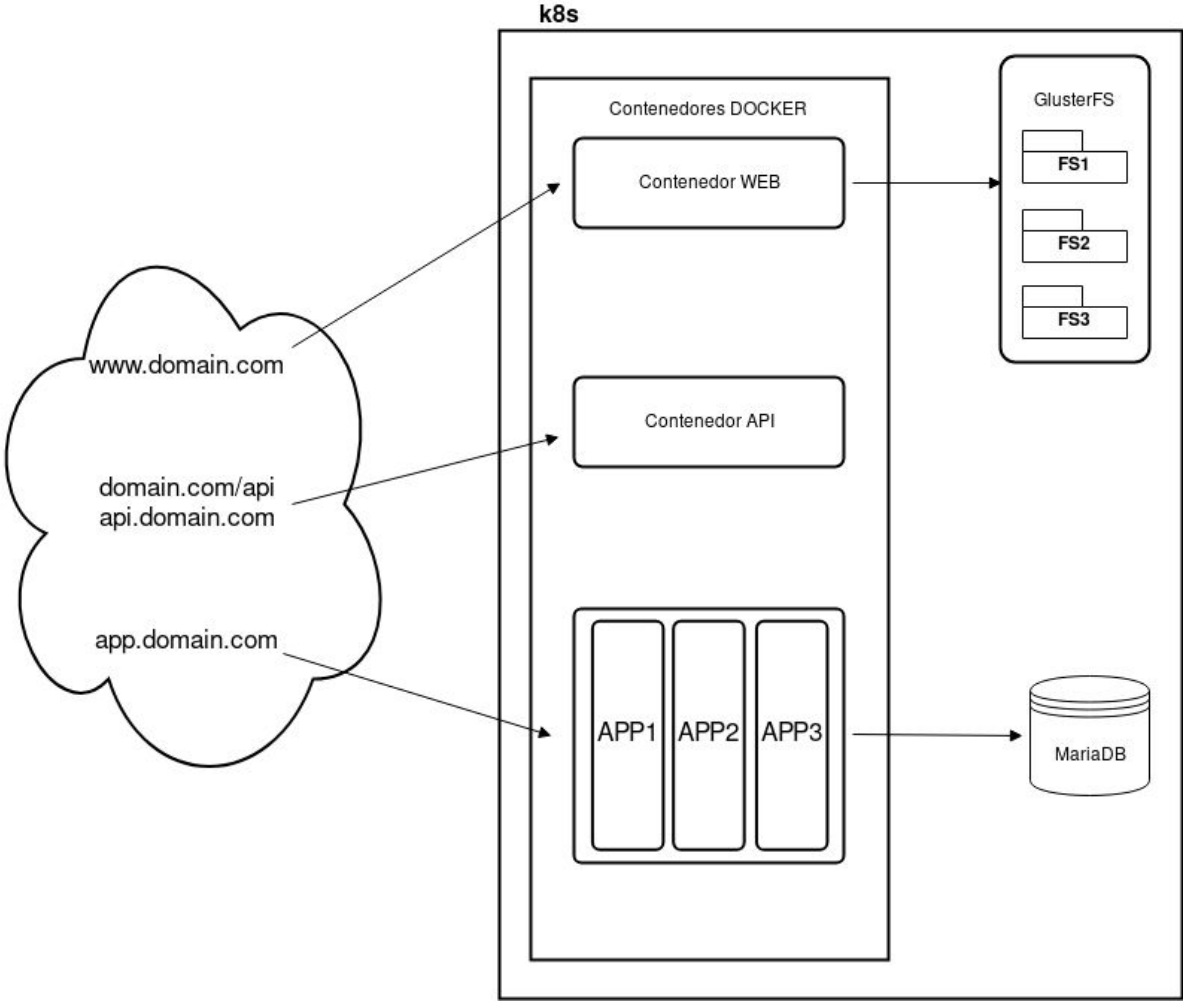
INTRODUCCIÓN

Objetivo

El proyecto que se pretende llevar a cabo trata de implementar un sistema de máquinas (virtuales o no) gestionadas por el software de orquestación Kubernetes (k8s), el proxy inverso Traefik.

De esta manera, se pretende lograr que estas máquinas sean accesibles a través de diferentes nombres de dominio o que la carga de trabajo quede distribuida entre ellas a través del balanceador de carga.

Para ello se ha optado por crear un entorno virtualizado gestionado por Vagrant, que trabaja sobre Virtualbox,



Herramientas

A continuación se enumeran las herramientas necesarias para poner en marcha la tarea planteada.

- Virtualbox
- Vagrant
- Docker
- Kubernetes
- Proxy inverso Traefik
- Vagrant-landrush

CONSIDERACIONES PREVIAS

OS base

El sistema base será un sistema operativo virtualizado sobre el que que se instalará el resto del software. Se ha escogido Ubuntu 16.04 LTS (xenial) por su sencillez y por la comunidad de usuarios que dispone, sobre la que poder apoyarse en caso de dificultades. Será necesario un entorno gráfico como GNome así como Docker para los gestión de contenedores que Kubernetes necesita, como ya se verá más adelante.

Para la gestión de la máquina virtual se ha optado por Vagrant junto con el plugin Landrush, que hará las funciones de servidor DNS, lo que evita que se tengan que definir estáticamente nombres de dominio.

Iteraciones

Cabe mencionar los intentos que se han realizado antes de escoger la implementación final de la tarea a realizar. Las guías de cada uno se pueden consultar en la bibliografía.

1. Minikube sobre virtualbox: dadas las limitaciones de los pc's del centro, no se pueden resolver dominios locales

2. Kubernetes multinodo, coreos: librerias deprecadas y/o no funcionales

3. Kubernetes multinodo, virtualizado con vagrant: complejidad muy alta o imposible

4. Kubernetes multinodo, virtualizado con vagrant: imposible levantar el pod para la red utilizando el plugin Flannel

5. Minikube no virtualizado, sobre un OS virtualizado con vagrant: se ha conseguido levantar minikube sobre un OS virtualizado a través de vagrant

IMPLEMENTACIÓN

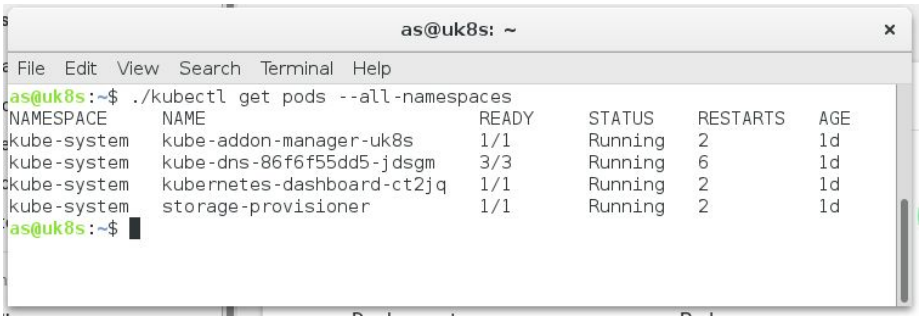
Implementación Kubernetes

Se va a utilizar la herramienta minikube que proporciona Kubernetes para la puesta en marcha de su clúster en modo local

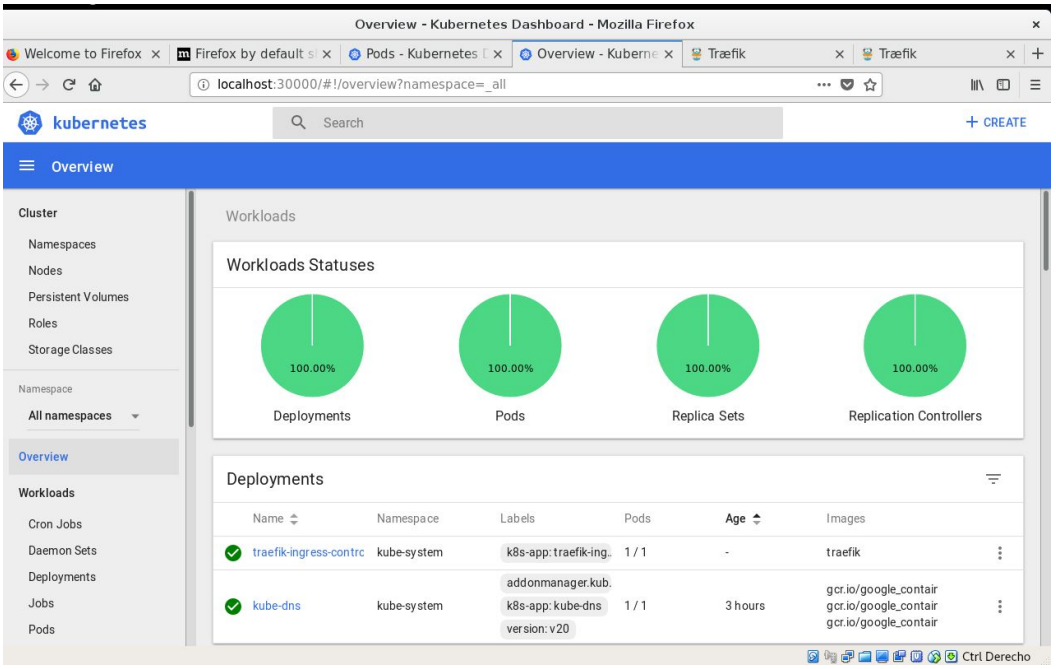
La instalación se reduce a la ejecución del siguiente script:

```
curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64 && chmod +x minikube
curl -Lo kubectl https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/
kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl && chmod +x kubectl
sudo -E ./minikube start --vm-driver=none
```

De este modo se instala lo necesario para tener el cluster en local, incluido un panel de control desde el que se pueden hacer todas la gestiones. Como el tiempo de descarga de todas las imágenes puede variar, se puede consultar en qué estado está con la siguiente orden:



Una vez terminado, se puede acceder al panel de control:

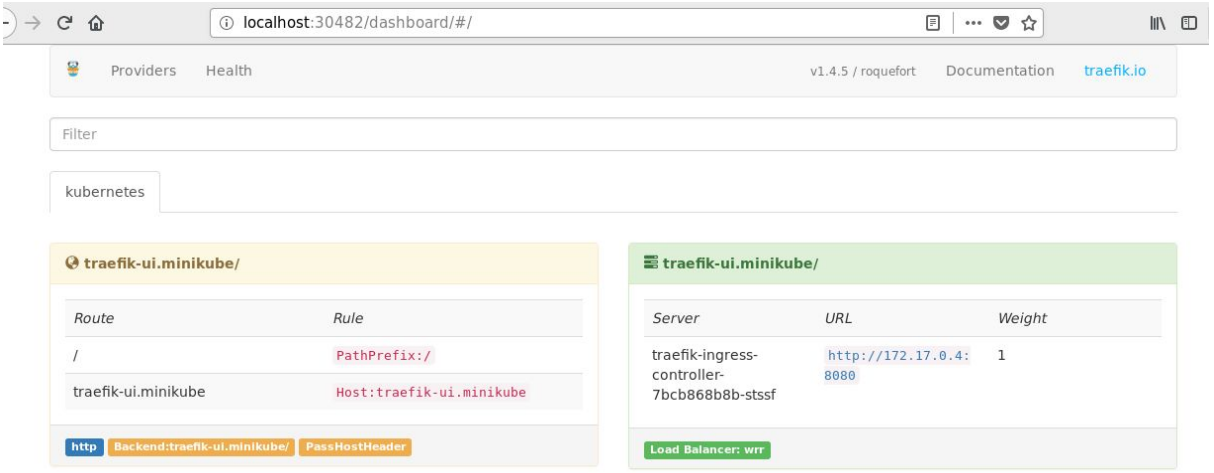


Implementación Traefik

La instalación del proxy inverso Traefik, también se reduce a la ejecución de las siguientes líneas de código:

```
# Deploy Træfik using a Deployment or DaemonSet
./kubectl apply -f https://raw.githubusercontent.com/containous/traefik/master/examples/k8s/traefik-deployment.yaml

# Submitting An Ingress to the cluster.
./kubectl apply -f https://raw.githubusercontent.com/containous/traefik/master/examples/k8s/ui.yaml
echo "$(minikube ip) traefik-ui.minikube" | sudo tee -a /etc/hosts
```

Es interesante mencionar la última línea de código que sirve para añadir una nueva entrada estática del nombre de dominio del panel de control de Traefik. Una vez descargado e instalado se podrá acceder a dicho panel de control.



Implementación: Traefik + Vagrant Landrush

Uno de los objetivos era no tener que añadir uno a uno entradas estáticas de nombres de dominio y para ello ya se ha adelantado que se iba a utilizar el plugin Landrush para Vagrant.

Para ello se debe habilitar el plugin vagrant-landrush dentro del fichero de configuración Vagrantfile:

```
  config.vm.provision :shell, :path => "setup.sh", :args =>
  config.landrush.enabled = true
  config.vm.define vm_name = "minikube" do |c|
      c.vm.hostname = vm_name
      config.landrush.tld = vm_name
      c.vm.network "private_network", ip: "192.168.99.100"
  end
```

Como nota interesante, cabe destacar que para resolver dinámicamente todos los subdominios, se asigna a la propiedad ‘tld’ el valor de hostname, de tal manera que se resolverán tanto el dominio hostname como los dominios del tipo subdominio.hostname y apuntar a nada IP indicada.

El plugin Landrush requiere definir una red privada. Este punto ha causado inconvenientes a la hora de levantar su correspondiente interfaz de red de la máquina virtual debido a que una de las últimas actualizaciones del sistema operativo provoca un mal funcionamiento. Para poder solventar este inconveniente se ha de ejecutar el siguiente código (véase bibliografía: workaround 1):

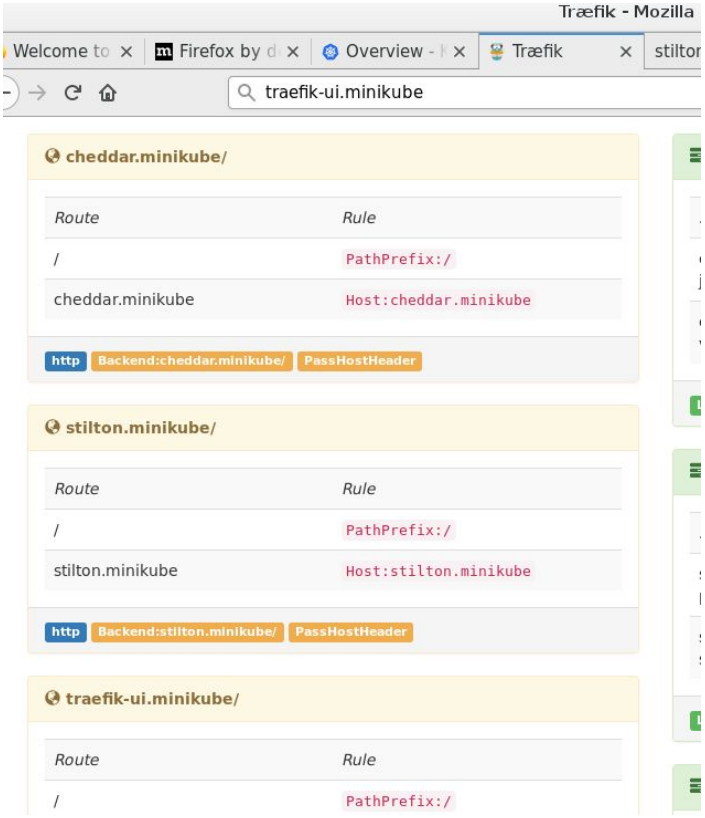
```
sudo mv /etc/network/fan /etc/network/fan.backup
```

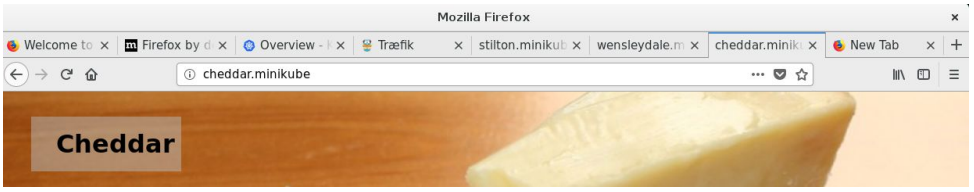
Por otro lado la definición de esta red privada provoca también problemas con la conectividad a los servidores de Docker con lo que impide la carga de la imagen de Traefik en este caso:

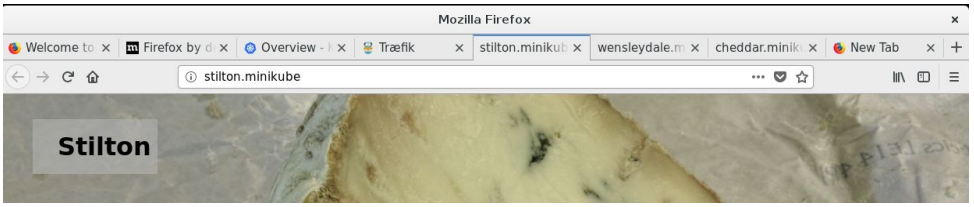
Para bordear esta circunstancia se debe ejecutar el siguiente código (véase bibliografía: workaround 2):

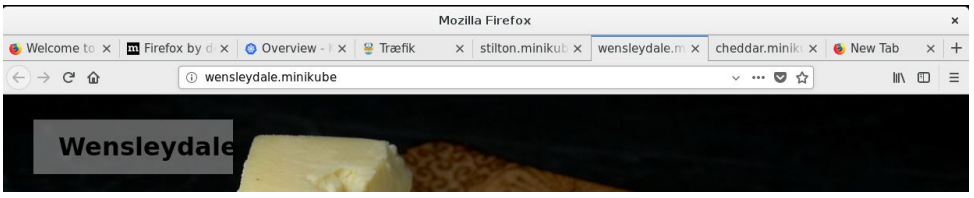
```
echo "$(host -t A index.docker.io | grep has.address | head -1 | awk '{print $NF}') index.docker.io" >> /etc/hosts

echo "$(host -t A registry-1.docker.io | grep has.address | head -1 | awk '{print $NF}') registry-1.docker.io" >> /etc/hosts
```

Una vez superados los obstáculos descritos se puede proceder a probar el funcionamiento del DNS que genera el plugin Landrush junto con el proxy inverso Traefik, para ello se implementan los ejemplos proporcionados por Traefik, teniendo en cuenta que ya no son necesarias las entradas estáticas de nombres de dominio, por lo que éstas se deben borrar del fichero hosts.









CONCLUSIONES

Dificultades para alcanzar el objetivo: tiempo, complejidad, bugs…
Técnicas apropiadas para entornos de desarrollo
Velocidad de evolución de las tecnologías (CI)
Complejidad de software de orquestación en local, alternativas.

POR HACER

Realizar una instalación del a versión de producción de kubernetes, añadir múltiples nodos al cluster en diferentes máquinas, tanto físicas como virtuales, poner a prueba el balanceador de carga mediante tests de carga.

MEJORAS

Localizar una utilidad para resolver el proxy DNS que funcione en conjunto con Traefik, en un entorno de producción y por lo tanto prescindiendo de Landrush.

BIBLIOGRAFÍA

- Kubernetes - https://kubernetes.io/docs/
- minikube sobre VM - https://kubernetes.io/docs/getting-started-guides/minikube/
- Cluster dind - https://github.com/Mirantis/kubeadm-dind-cluster
- minikube sobre VM - https://www.stratoscale.com/blog/kubernetes/installing-kubernetes-bare-metal/
- minikube no VM - https://github.com/kubernetes/minikube
- Vagrant - https://kubernetes.io/
- Servicio Traefik - https://www.praqma.com/stories/proxying-kubernetes/
- Traefik sobre k8s - https://docs.traefik.io/user-guide/kubernetes/
- Landrush - https://github.com/vagrant-landrush/landrush

Workaround

1. Bug red privada en Ubuntu: https://bugs.launchpad.net/ubuntu/+source/ubuntu-fan/+bug/1729608
2. Bug bloqueo de red docker: https://github.com/vagrant-landrush/landrush/issues/293

Repositorios

- GitHub - https://github.com/eaguirrezaba001/as
- Vagrant Cloud - https://app.vagrantup.com/eaguirrezaba001/boxes/as