TALLER SISTEMAS OPERATIVOS:
Kubeclt, Minikube y Kubernetes.

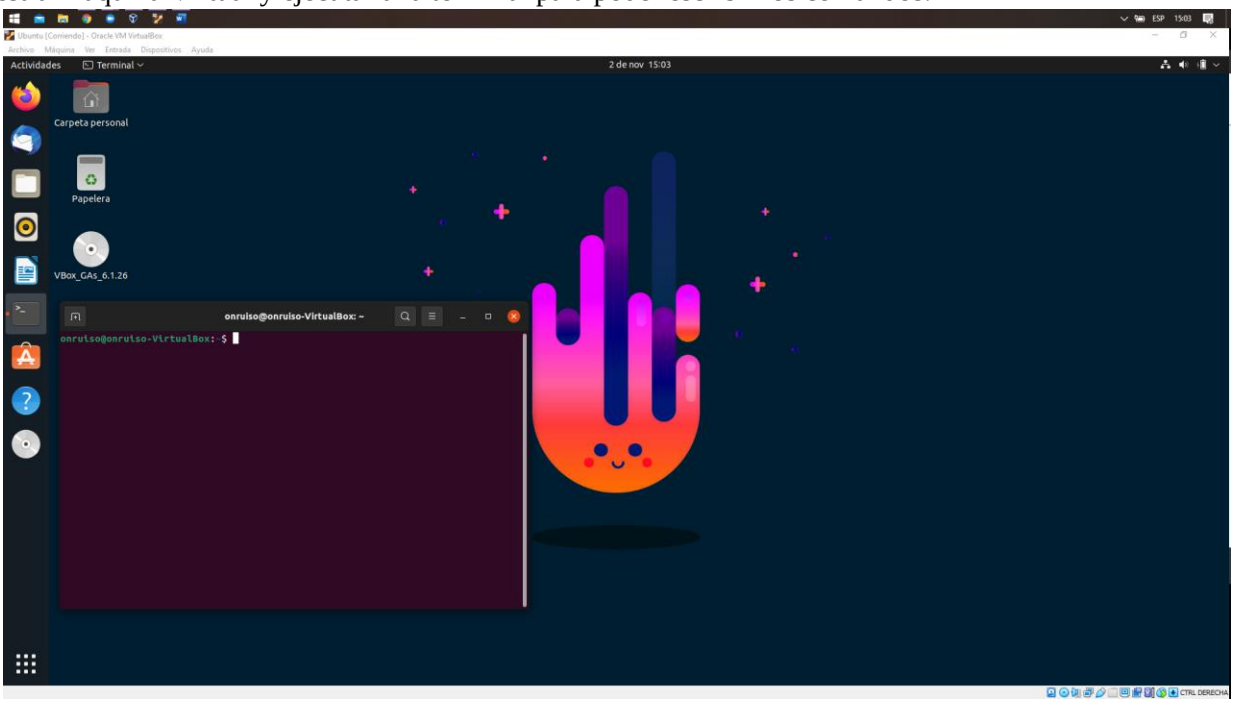
Luis Felipe Narváez Gómez. E-mail: luis.narvaez@usantoto.edu.co. Cod: 2312660. Facultad de Ingeniería de Sistemas.

INSTALAR KUBECTL

Para instalar Kubectl en Linux, se recomienda utilizar la distribución de Ubuntu o semejantes. Utilizar distribuciones de uso especifico como las de seguridad informática como BlackArch o Kali Linux presenta, varios problemas con el buen funcionamiento e instalación de paquetes y librerías que necesarias.
Antes de poder instalar Kubectl se recomienda instalar una versión menor al cluster que se tenga instalado, es decir, si el cliente con el que se esta trabajando es el V1.22, este podrá comunicarse con la versión v1.21, v1.22 y v1.23; esto ayudara a evitar problemas e imprevistos en la instalación.
La guía oficial de Instalación de Kubectl se puede encontrar en la página web de Kubernetes, tanto para Sistemas Operativos Linux, macOS y Windows. Este se puede encontrar en el link *https://kubernetes.io/docs/tasks/tools/#install-kubectl-on-linux* . En esta Guia utilizaremos la sección orientada a las distribuciones de Linux, la cual se halla en el link *https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/* . La instalación en Linux puede hacerse de tres maneras diferentes:

1. A partir de Binarios con el recurso de CURL en Linux.
2. Usando la administración de paquetes nativa.
3. Usando otra administración de paquetes.

En esta guía utilizaremos la primera opción, utilizando los binarios del recuro CURL de Ubuntu Linux. Lo primero será abrir Nuestra Maquina Virtual y ejecutar una terminal para poder escribir los comandos.



Dentro de la consola de comandos o Terminal de Ubuntu, descargaremos la última versión de Kubectl con el siguiente comando.

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
```

Puesto en práctica tendríamos:

```
onruiso@onruiso-VirtualBox:~$ sudo curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
[sudo] contraseña para onruiso:
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   154  100   154    0     0    131      0  0:00:01  0:00:01 --:--:--   131
100 44.7M  100 44.7M    0     0  1915k      0  0:00:23  0:00:23 --:--:-- 3949k
onruiso@onruiso-VirtualBox:~$
```

En caso de que se haya querido una versión especifica, debemos reemplazar la siguiente línea del comando, por la versión especifica.

$(curl -L -s https://dl.k8s.io/release/stable.txt)

Por ejemplo, en caso de querer la versión v1.22.0 en Linux, el comando se transformaría a:

curl -LO https://dl.k8s.io/release/v1.22.0/bin/linux/amd64/kubectl

Ahora, debemos validar el binario, para eso, primero debemos descargar el archivo adicional de comprobación de Kubetcl (Esta parte de por si es opcional, pero se recomienda hacer). El comando es:

curl -LO "https://dl.k8s.io/**$(**curl -L -s https://dl.k8s.io/release/stable.txt**)**/bin/linux/amd64/kubectl.sha256"

Puesto en práctica tendríamos:

```
onruiso@onruiso-VirtualBox:~$ sudo curl -LO "https://dl.k8s.io/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   154  100   154    0     0    209      0 --:--:-- --:--:-- --:--:--   209
100    64  100    64    0     0     30      0  0:00:02  0:00:02 --:--:--    63
onruiso@onruiso-VirtualBox:~$
```

Ahora podemos validar el binario con el archivo adicional de comprobación que utilizamos con el siguiente comando:

echo "**$(**<kubectl.sha256**)** kubectl" | sha256sum –check

Poniéndolo en práctica, témenos:

```
onruiso@onruiso-VirtualBox:~$ sudo echo "$(<kubectl.sha256) kubectl" | sha256sum --check
kubectl: La suma coincide
onruiso@onruiso-VirtualBox:~$
```

También puede salir algún mensaje como “kubectl: OK”, sin embargo, en cado de que la verificación falle, “sha256” saldrá con un valor distinto a cero y /o imprime una salida similar a lo siguiente:

kubectl: FAILED
sha256sum: WARNING: 1 computed checksum did NOT match

En caso de esto último, se debe volver a descargar la misma versión del binario y así la misma versión del paquete adicional de comprobación. Ahora bien, para instalar Kubectl como tal, utilizaremos el siguiente comando:

sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

Poniéndolo en práctica, témenos:

```
onruiso@onruiso-VirtualBox:~$ sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
onruiso@onruiso-VirtualBox:~$
```

En caso de que no se tenga el acceso a root en el sistema que estemos trabajando, aun podemos instalar Kubectl en la ruta “~/.local/bin” de la siguiente forma:

chmod +x kubectl
mkdir -p ~/.local/bin/kubectl
mv ./kubectl ~/.local/bin/kubectl
 # and then add ~/.local/bin/kubectl to $PATH

Una vez instalado podemos comprobar la versión que hemos instalado de Kubectl con el siguiente comando:

```
kubectl version –client
```

Puesto en práctica, tenemos:

INSTALAR MINIKUBE START

El MiniKube es una versión de Kubernetes pero de forma Local a nuestra Maquina Virtual, la cual esta orientada en facilitar el aprendizaje y el desarrollo para Kubernets, para este, solo necesiamos el contenedor Docker o uno similar para que sea compatible o bien un entorno de Máquina Virtual. Para Usar MiniKube es necesario cumplir una serie de requisitos en nuestra Máquina Virtual, estos son:

1. 2 CPU o más
2. 2GB de memoria libre
3. 20 GB de espacio libre en disco
4. conexión a Internet

Administrador de contenedores o máquinas virtuales, como: Docker, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox o VMware
Una vez nos hemos asegurado que nuestra Maquina Virtual cumpla con los requisitos para funcionar MiniKube, podemos Iniciar la Instalación. Para Poder Instalar Nuestro MiniKube debemos tener en cuenta ciertos Parámetros u opciones de instalación que queremos utilizar, esto deviene de la herramienta de La guía oficial de MiniKube para su instalación, la cual genera el código o comando necesario para poder instalar la utilidad.

En este caso, la máquina en donde Instalaremos MiniKube es Ubuntu Linux, utilizaremos la última versión estable del sistema e instalaremos una versión de Binarios CURL. Para poder ver la Arquitectura de nuestra máquina virtual podremos utilizar alguno de los siguientes dos comandos:

Sudo uname -m
Sudo uname -a

Poniéndolos en práctica tenemos:

```
onruiso@onruiso-VirtualBox:~$ sudo uname -m
x86_64
onruiso@onruiso-VirtualBox:~$ sudo uname -a
Linux onruiso-VirtualBox 5.11.0-38-generic #42-Ubuntu SMP Fri Sep 24 14:03:54 UT
C 2021 x86_64 x86_64 x86_64 GNU/Linux
onruiso@onruiso-VirtualBox:~$
```

De esta manera podemos saber la arquitectura de nuestra maquina con seguridad y así obtener para MI CASO la siguiente línea de comandos:

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube

Poniéndolo en práctica tendríamos:

```
onruiso@onruiso-VirtualBox:~$ sudo curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 65.8M  100 65.8M    0     0   440k      0  0:02:33  0:02:33 --:--:-- 3684k
onruiso@onruiso-VirtualBox:~$
```

```
onruiso@onruiso-VirtualBox:~$ sudo install minikube-linux-amd64 /usr/local/bin/minikube
onruiso@onruiso-VirtualBox:~$
```

Ahora bien, podemos iniciar el Cluster, para lo cual utilizaremos el siguiente comando:

minikube start

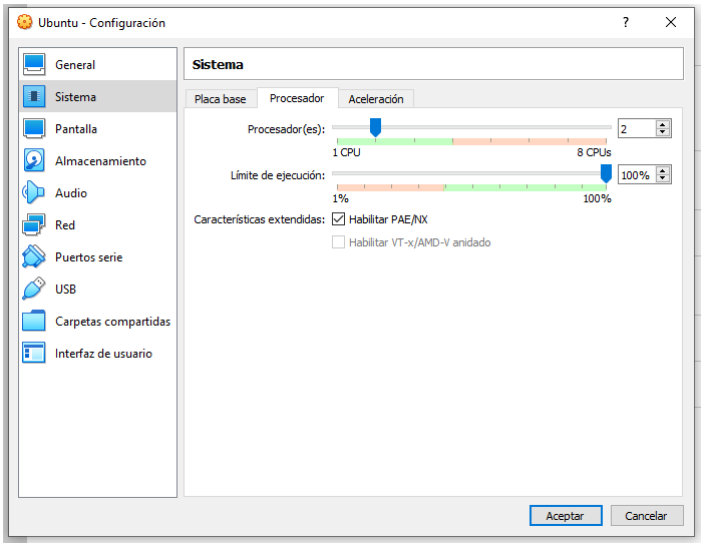
Poniéndolo en práctica tenemos:

```
onruiso@onruiso-VirtualBox:~$ sudo minikube start
😄  minikube v1.23.2 en Ubuntu 21.04 (vbox/amd64)
✨  Controlador docker seleccionado automáticamente. Otras opciones: ssh, none

⛔  Exiting due to RSRC_INSUFFICIENT_CORES: Requested cpu count 2 is greater than the available cpus of 1

onruiso@onruiso-VirtualBox:~$
```

Como podemos ver en la ejecución del comando, hemos incumplido uno de los requisitos mínimos de funcionamiento de MiniKube, este es que nuestro sistema debe tener 2 CPU de uso mínimo. Si se tratara de una Maquina Real, se debe de cambiar el PROCESADOR de nuestra computadora, una que por lo mínimo tenga 2 núcleos de procesamiento; en este caso, al tratarse de una Maquina Virtual, podemos cambiar el numero de CPU de la siguiente forma (Se esta utilizando el software de Virtual Box).
Dentro de Virtual Box y con la Maquina Virtual apagada, seleccionaremos la misma y daremos en la opción de configuración.
Dentro de la ventana desplegada iremos a la barra lateral izquierda y seleccionaremos “sistema”, dentro de la cual iremos a la pestaña de “Procesador” y aumentaremos el numero de procesadores de nuestra Maquina virtual. Debemos tener en cuenta que aun siendo una Maquina virtual, tanto la cantidad de RAM como la de CPU dadas a la misma, serán valores reales que la Maquina Anfitriona le entregara para ser utilizados, esto quiere decir por ejemplo que, si tenemos un uso de 1 núcleo y 1GB de RAM normalmente en nuestra PC, al iniciar la maquina virtual, los recursos utilizados serán de 3 núcleos y 5GB de RAM, suponiendo que la RAM entregada a la Maquina Virtual sea de 4GB.



Una vez hecho esto, volvemos a iniciar la Maquina Virtual y ejecutamos el comando “sudo minikube start”, por lo cual tenemos ahora:

```
onruiso@onruiso-VirtualBox:~$ sudo minikube start
[sudo] contraseña para onruiso:
😄  minikube v1.23.2 en Ubuntu 21.04 (vbox/amd64)
✨  Controlador docker seleccionado automáticamente. Otras opciones: none, ssh
🎈  The "docker" driver should not be used with root privileges.
💡  If you are running minikube within a VM, consider using --driver=none:
📘    https://minikube.sigs.k8s.io/docs/reference/drivers/none/

❌  Exiting due to DRV_AS_ROOT: The "docker" driver should not be used with root privileges.

onruiso@onruiso-VirtualBox:~$
```

Para resolver este error es comprobando varios aspectos durante la instalación de Minikube. Para eso utilizaremos la guía de instalación de MinKube de Phoenix Nap Global IT Services, la cual podemos ver en el enlace *https://phoenixnap.com/kb/install-minikube-on-ubuntu* .
Lo primero que debemos hacer es comprobara que las versiones del software y registros del sistema esten actualizados, para ello utilizaremos los comando de:

```
sudo apt-get update -y
sudo apt-get upgrade -y
```

Puesto en Practica tenemos al primer comando:

```
onruiso@onruiso-VirtualBox:~$ sudo apt-get update -y
[sudo] contraseña para onruiso:
Obj:1 http://co.archive.ubuntu.com/ubuntu hirsute InRelease
Des:2 http://security.ubuntu.com/ubuntu hirsute-security InRelease [110 kB]
Des:3 http://co.archive.ubuntu.com/ubuntu hirsute-updates InRelease [115 kB]
Des:4 https://download.docker.com/linux/ubuntu hirsute InRelease [48,9 kB]
Des:5 http://co.archive.ubuntu.com/ubuntu hirsute-backports InRelease [101 kB]
Des:6 http://co.archive.ubuntu.com/ubuntu hirsute-updates/main i386 Packages [204 kB]
Des:7 http://security.ubuntu.com/ubuntu hirsute-security/main amd64 Packages [279 kB]
Des:8 http://co.archive.ubuntu.com/ubuntu hirsute-updates/main amd64 Packages [409 kB]
Des:9 http://co.archive.ubuntu.com/ubuntu hirsute-updates/main Translation-en [108 kB]
Des:10 http://co.archive.ubuntu.com/ubuntu hirsute-updates/main amd64 DEP-11 Metadata [95,0 kB]
Des:11 http://co.archive.ubuntu.com/ubuntu hirsute-updates/main amd64 c-n-f Metadata [7.848 B]
Des:12 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe i386 Packages [243 kB]
Des:13 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe amd64 Packages [339 kB]
Des:14 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe Translation-en [83,9 kB]
Des:15 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe amd64 DEP-11 Metadata [57,9 kB]
Des:16 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe amd64 c-n-f Metadata [7.736 B]
Des:17 http://co.archive.ubuntu.com/ubuntu hirsute-updates/multiverse amd64 DEP-11 Metadata [944 B]
Des:18 http://co.archive.ubuntu.com/ubuntu hirsute-backports/universe amd64 DEP-11 Metadata [9.348 B]
Des:19 http://security.ubuntu.com/ubuntu hirsute-security/main i386 Packages [113 kB]
Des:20 http://security.ubuntu.com/ubuntu hirsute-security/main Translation-en [71,6 kB]
Des:21 http://security.ubuntu.com/ubuntu hirsute-security/main amd64 DEP-11 Metadata [9.708 B]
Des:22 http://security.ubuntu.com/ubuntu hirsute-security/main amd64 c-n-f Metadata [4.692 B]
Des:23 http://security.ubuntu.com/ubuntu hirsute-security/universe amd64 Packages [225 kB]
Des:24 http://security.ubuntu.com/ubuntu hirsute-security/universe i386 Packages [195 kB]
Des:25 http://security.ubuntu.com/ubuntu hirsute-security/universe Translation-en [47,9 kB]
Des:26 http://security.ubuntu.com/ubuntu hirsute-security/universe amd64 DEP-11 Metadata [5.664 B]
Des:27 http://security.ubuntu.com/ubuntu hirsute-security/universe amd64 c-n-f Metadata [5.468 B]
Descargados 2.898 kB en 3s (902 kB/s)
Leyendo lista de paquetes... Hecho
onruiso@onruiso-VirtualBox:~$
```

Y al segundo comando como:

```
onruiso@onruiso-VirtualBox:~$ sudo apt-get upgrade -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Calculando la actualización... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  libllvm11
Utilice «sudo apt autoremove» para eliminarlo.
Se actualizarán los siguientes paquetes:
```

…

```
Procesando disparadores para desktop-file-utils (0.26-1ubuntu1) ...
Procesando disparadores para hicolor-icon-theme (0.17-2) ...
Procesando disparadores para gnome-menus (3.36.0-1ubuntu1) ...
Procesando disparadores para libc-bin (2.33-0ubuntu5) ...
Procesando disparadores para man-db (2.9.4-2) ...
Procesando disparadores para dbus (1.12.20-1ubuntu3) ...
Procesando disparadores para mailcap (3.68ubuntu1) ...
onruiso@onruiso-VirtualBox:~$
```

Una vez hemos hecho esto comprobaremos que de verdad tenemos instalado el paquete con el cual podemos instalar un software desde binarios y el de transporte por https. Esto se realiza con los siguientes comandos:

```
sudo apt-get install curl
sudo apt-get install apt-transport-https
```

Puesto en práctica el primer comando, tenemos lo siguiente:

```
onruiso@onruiso-VirtualBox:~$ sudo apt-get install curl
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
curl ya está en su versión más reciente (7.74.0-1ubuntu2.3).
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  libllvm11
Utilice «sudo apt autoremove» para eliminarlo.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
onruiso@onruiso-VirtualBox:~$
```

Como podemos observar al ejecutar el primer comando, en este caso si tenemos instalado y actualizado el paquete de CURL, sin embargo, como vemos más adelante, no poseíamos el paquete de TRANSPORT HTTPS.

```
onruiso@onruiso-VirtualBox:~$ sudo apt-get install apt-transport-https
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  libllvm11
Utilice «sudo apt autoremove» para eliminarlo.
Se instalarán los siguientes paquetes NUEVOS:
  apt-transport-https
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 4.776 B de archivos.
Se utilizarán 166 kB de espacio de disco adicional después de esta operación.
Des:1 http://co.archive.ubuntu.com/ubuntu hirsute-updates/universe amd64 apt-transport-https all 2.2.
4ubuntu0.1 [4.776 B]
Descargados 4.776 B en 1s (7.272 B/s)
Seleccionando el paquete apt-transport-https previamente no seleccionado.
(Leyendo la base de datos ... 199651 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../apt-transport-https_2.2.4ubuntu0.1_all.deb ...
Desempaquetando apt-transport-https (2.2.4ubuntu0.1) ...
Configurando apt-transport-https (2.2.4ubuntu0.1) ...
onruiso@onruiso-VirtualBox:~$
```

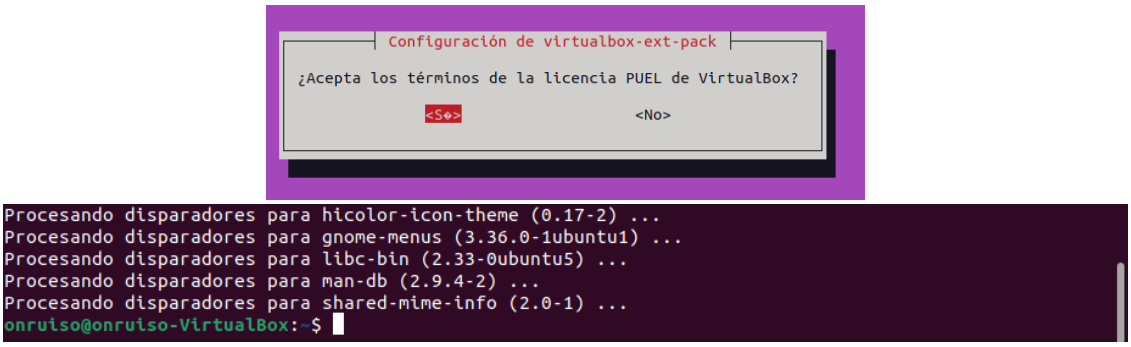
Ahora bien, instalaremos el Hipervisor de VirtualBox. Para poder configurar un cluster, del cual hará uso minikube, necesitamos una máquina virtual en la que se pueda configurar el nombrado cluster de nodo único, esto con el mismo MiniKube. Para esto podemos utilizar cualquier Maquina Virtual, siendo en este caso Virtual Box.
Para instalar Virtual Box dentro de Ubuntu, podemos utilizar el siguiente comando:

```
sudo apt install virtualbox virtualbox-ext-pack
```

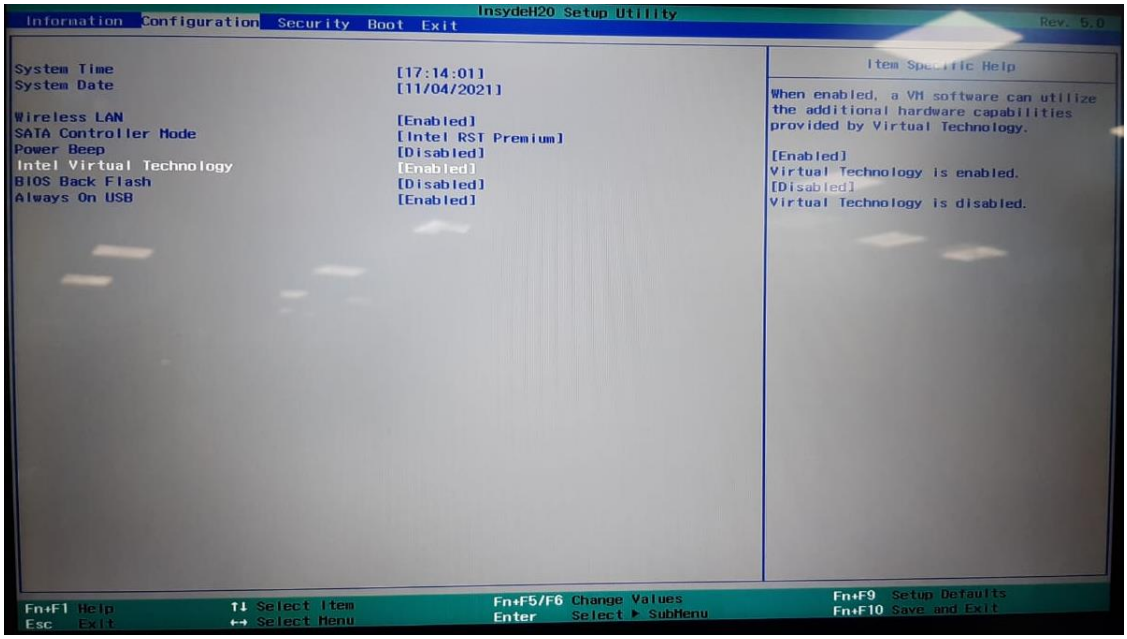
Poniendo en practica el comando tenemos el siguiente escenario:

```
onruiso@onruiso-VirtualBox:~$ sudo apt install virtualbox virtualbox-ext-pack
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  libllvm11
Utilice «sudo apt autoremove» para eliminarlo.
Se instalarán los siguientes paquetes adicionales:
  dctrl-tools dkms libdouble-conversion3 libgsoap-2.8.104 liblzf1 libmd4c0 libpcre2-16-0
```

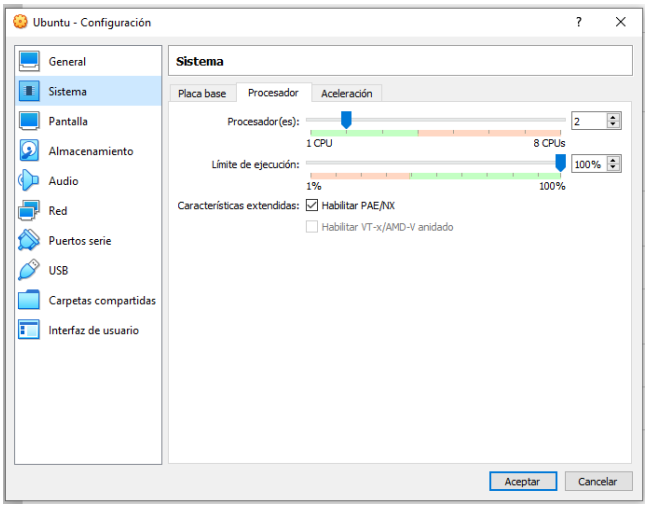
A mitad de la instalación, nos aparecerá una pantalla en la que confirmaremos el contrato de licencia de cirtual box. Aceptaremos el contrato y sus términos. Luego de dar estas indicaciones podremos continuar con la instalación de forma corriente.



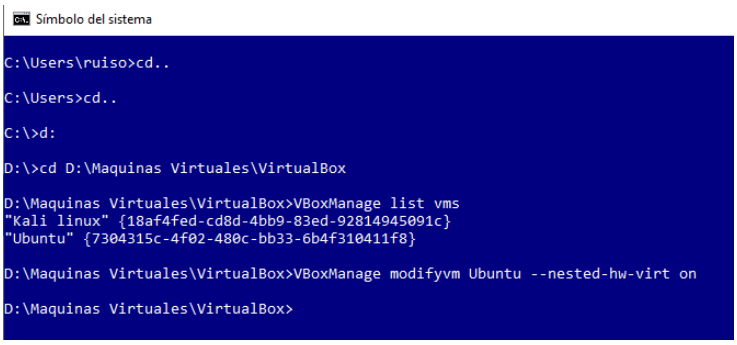
Algo que debemos tener en cuenta, es que para que este HIPERVISOR VIRTUAL BOX funcione correctamente, la virtualización de hardware debe estar habilitada en la BIOS de la Maquina Anfitrión. Para asegurarnos daremos una visita a nuestra BIOS donde pondremos la virtualización en Enable.



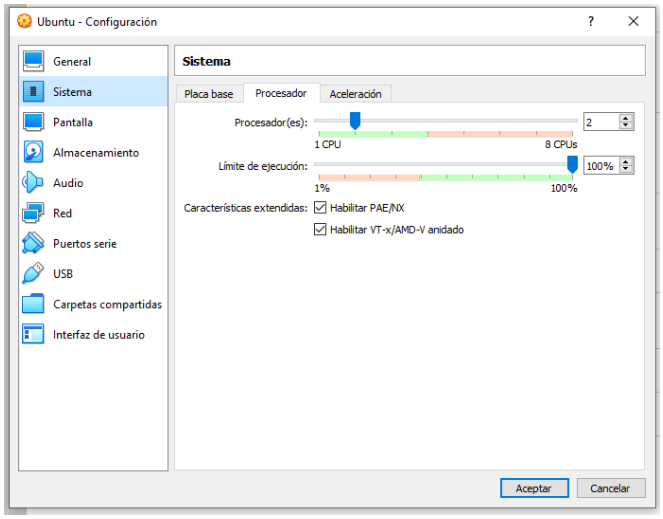
Una vez hecho esto comprobaremos que este habilitado en virtual box.



En aso de figurar que no se puede marcar “VT-x/AMD-V anidado”, procederemos con los siguientes comandos en nuestra Maquina anfitrión.



Una vez ejecutado estos comandos podemos regresar a Virtual Box y comprobar que ya está habilitada la opción que queremos.



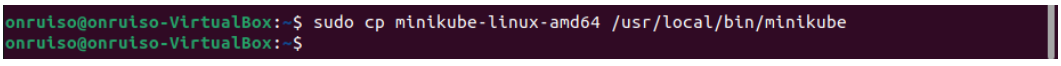
Ahora si podemos volver a instalar Minikube, para ello utilizaremos el siguiente comando de Linux:

```
sudo wget https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
```

La Ejecución del comando en la terminal es la siguiente:

Seguido de esto, copiaremos el archivo que se acaba de descargar y guardaremos en el directorio de minikube, esto con el siguiente comando:

```
sudo cp minikube-linux-amd64 /usr/local/bin/minikube
```



Ahora otorgaremos permiso al ejecutivo de archivos mediante el comando de CHMOD.

```
sudo chmod 755 /usr/local/bin/minikube
```

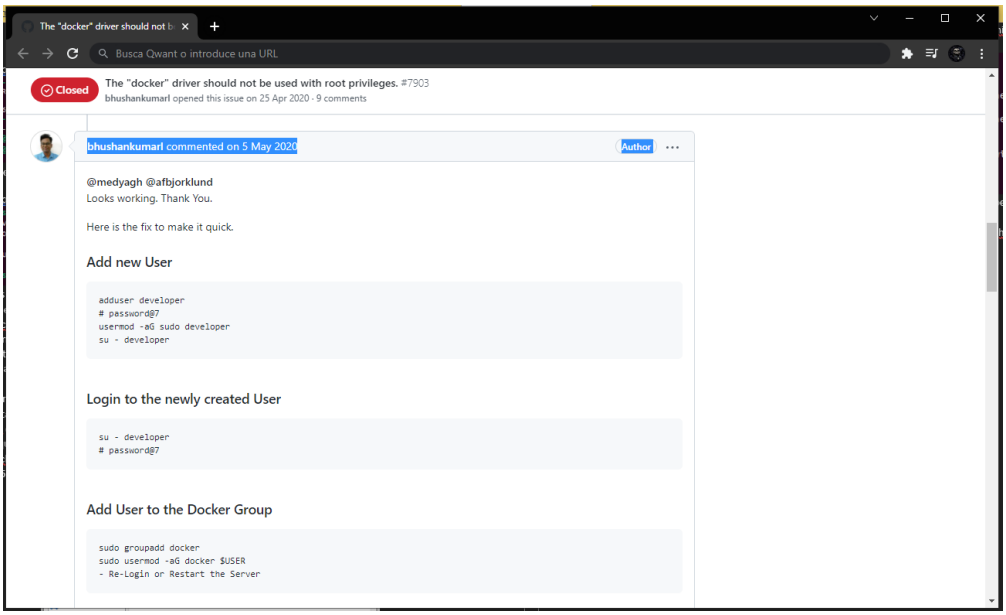
```
onruiso@onruiso-VirtualBox:~$ sudo chmod 755 /usr/local/bin/minikube
onruiso@onruiso-VirtualBox:~$
```

De esta forma si podremos ver la versión de nuestro minikube con el comando:

```
minikube version
```

```
onruiso@onruiso-VirtualBox:~$ minikube version
minikube version: v1.23.2
commit: 0a0ad76465208247 7c00d51d2475284b5d39ceed
onruiso@onruiso-VirtualBox:~$
```

Ahora solucionemos el error específico del ROOT, el cual nos dice que el “Docker” no debería utilizarse con privilegios de ROOT. Para solucionar este error, podemos revisar el siguiente enlace *https://github.com/kubernetes/minikube/issues/7903* , concretamente el comentario de “bhushankumarl” dado en la fecha del 5 de mayo del año 2020.



En este post podremos recibir una serie de comandos para poder utilizar un nuevo usuario.
Añadir una nuevo Usuario, sus comandos son:

```
adduser developer
```

Si lo ponemos en práctica tenemos:

```
onruiso@onruiso-VirtualBox:~$ sudo adduser developer
[sudo] contraseña para onruiso:
Añadiendo el usuario `developer' ...
Añadiendo el nuevo grupo `developer' (1001) ...
Añadiendo el nuevo usuario `developer' (1001) con grupo `developer' ...
El directorio personal `/home/developer' ya existe.  No se copiará desde `/etc/skel'.
Nueva contraseña:
```

```
#password@7
```

```
Vuelva a escribir la nueva contraseña:
```

#password@7

```
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para developer
Introduzca el nuevo valor, o presione INTRO para el predeterminado
        Nombre completo []:
```

INTRO

```
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para developer
Introduzca el nuevo valor, o presione INTRO para el predeterminado
        Nombre completo []:
        Número de habitación []:
        Teléfono del trabajo []:
        Teléfono de casa []:
        Otro []:
¿Es correcta la información? [S/n] S
onruiso@onruiso-VirtualBox:~$
```

sudo usermod -aG sudo developer

```
onruiso@onruiso-VirtualBox:~$ sudo usermod -aG sudo developer
onruiso@onruiso-VirtualBox:~$
```

Ahora iniciamos sesión con el nuevo usuario recién creado, con el siguiente comando:

sudo su - developer

```
onruiso@onruiso-VirtualBox:~$ sudo su - developer
developer@onruiso-VirtualBox:~$
```

Agregaremos el nuevo usuario al grupo de Docker con los siguientes comandos:

sudo groupadd docker

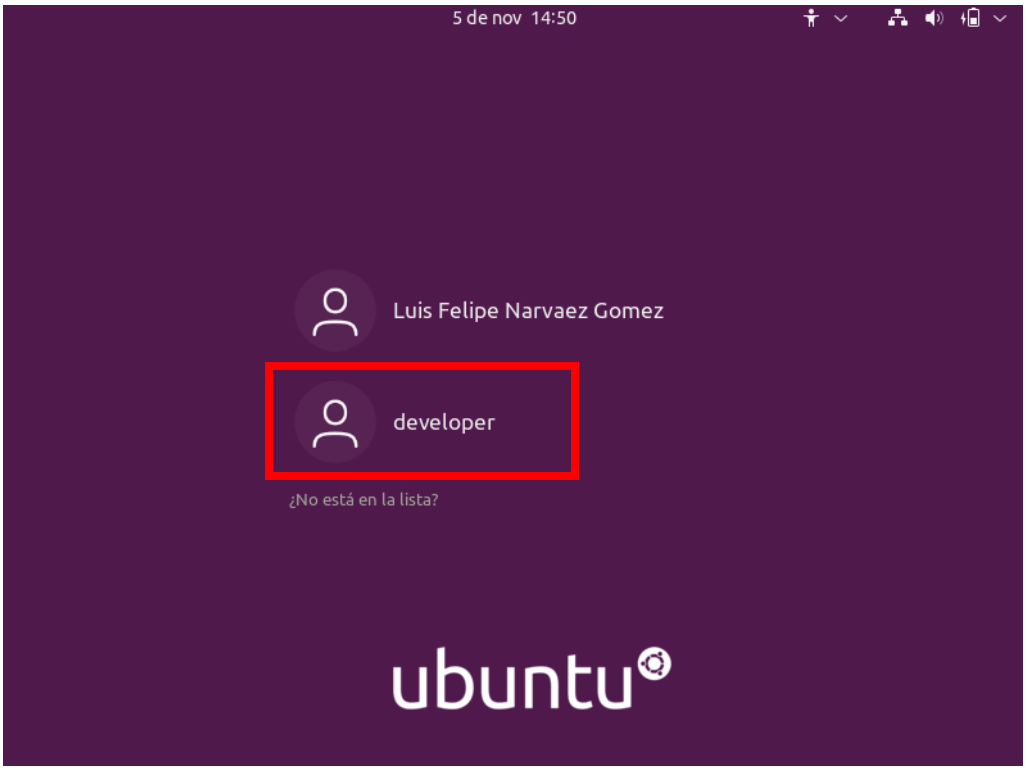
```
developer@onruiso-VirtualBox:~$ sudo groupadd docker
[sudo] contraseña para developer:
Lo sentimos, vuelva a intentarlo.
[sudo] contraseña para developer:
groupadd: el grupo «docker» ya existe
developer@onruiso-VirtualBox:~$
```

En el anterior comando creábamos el grupo Docker, sin embargo, esta acción no se aplica pues, el mismo ya existe.

sudo usermod -aG docker $USER

```
developer@onruiso-VirtualBox:~$ sudo usermod -aG docker $USER
developer@onruiso-VirtualBox:~$
```

- Re-Login or Restart the Server Reiniciamos la maquina virtual y accedemos al nuevo usuario que hemos creado.



Reinstalemos Minikube con el siguiente comando.

sudo curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
developer@onruiso-VirtualBox:~$ sudo curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 66.3M  100 66.3M    0     0  16.6M      0  0:00:03  0:00:03 --:--:-- 16.6M
developer@onruiso-VirtualBox:~$
```

chmod +x minikube

```
developer@onruiso-VirtualBox:~$ sudo chmod +x minikube
developer@onruiso-VirtualBox:~$
```

sudo mv ./minikube /usr/local/bin/minikube

```
developer@onruiso-VirtualBox:~$ sudo mv ./minikube /usr/local/bin/minikube
developer@onruiso-VirtualBox:~$
```

Iniciamos Minikube con el Driver de Docker con el siguiente comando.

minikube start --driver=docker

```
developer@onruiso-VirtualBox:~$ minikube start --driver=docker
😄  minikube v1.24.0 en Ubuntu 21.04
✨  Using the docker driver based on user configuration
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
💾  Descargando Kubernetes v1.22.3 ...
    > preloaded-images-k8s-v13-v1...: 501.73 MiB / 501.73 MiB  100.00% 11.13 Mi
    > gcr.io/k8s-minikube/kicbase: 355.77 MiB / 355.78 MiB  100.00% 6.80 MiB p/
🔥  Creando docker container (CPUs=2, Memory=2200MB) ...
🐳  Preparando Kubernetes v1.22.3 en Docker 20.10.8...
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎  Verificando Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Complementos habilitados: default-storageclass, storage-provisioner
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
developer@onruiso-VirtualBox:~$
```

Verificar la instalación de minikube.

docker ps

```
developer@onruiso-VirtualBox:~$ sudo docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED         ST
ATUS         PORTS
                                                    NAMES
27f164f4585a   gcr.io/k8s-minikube/kicbase:v0.0.28   "/usr/local/bin/entr…"   4 minutes ago   Up
 4 minutes   127.0.0.1:49157->22/tcp, 127.0.0.1:49156->2376/tcp, 127.0.0.1:49155->5000/tcp, 127.
0.0.1:49154->8443/tcp, 127.0.0.1:49153->32443/tcp   minikube
developer@onruiso-VirtualBox:~$
```

Listo, ahora tenemos MINIKUBE en nuestra Máquina Virtual.

ADMINISTRAR KUBERNETES CON MINIKUBE

Ver la configuración actual de KUBECTL.

kubectl config view

```
developer@onruiso-VirtualBox:~$ sudo kubectl config view
apiVersion: v1
clusters: null
contexts: null
current-context: ""
kind: Config
preferences: {}
users: null
developer@onruiso-VirtualBox:~$
```

Los siguientes comandos no nos darán información en este momento de ejecución, esto debido a que no tenemos CLUSTER creados todavía, sin embargo, se demuestra su ejecución en Ubuntu. Mostrar la información del cluster.

```
developer@onruiso-VirtualBox:~$ minikube start
😄  minikube v1.24.0 en Ubuntu 21.04
✨  Using the docker driver based on existing profile
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
🏃  Updating the running docker "minikube" container ...
🐳  Preparando Kubernetes v1.22.3 en Docker 20.10.8...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Complementos habilitados: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
developer@onruiso-VirtualBox:~$
```

kubectl cluster-info

```
developer@onruiso-VirtualBox:~$ sudo kubectl cluster-info

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
The connection to the server localhost:8080 was refused - did you specify the right host or port
?
developer@onruiso-VirtualBox:~$ sudo kubectl cluster-info dump
The connection to the server localhost:8080 was refused - did you specify the right host or port
?
developer@onruiso-VirtualBox:~$
```

Verificar los nodos en ejecución.

kubectl get nodes

```
developer@onruiso-VirtualBox:~$ kubectl get nodes
NAME       STATUS   ROLES                  AGE   VERSION
minikube   Ready    control-plane,master   24m   v1.22.3
developer@onruiso-VirtualBox:~$
```

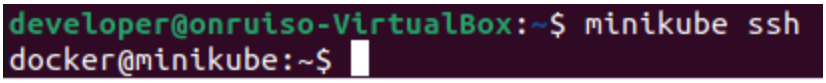
Para ver una lista de todos los PODS de MINIKUBE ejecutados.

kubectl get pod

```
developer@onruiso-VirtualBox:~$ sudo kubectl get pod
The connection to the server localhost:8080 was refused - did you specify the right host or port
?
developer@onruiso-VirtualBox:~$
```

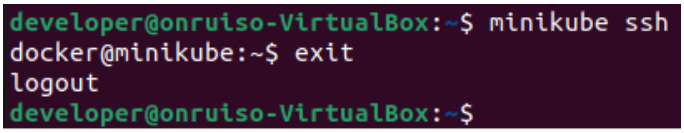
Para ingresar a la VM de Minikube.

```
minikube ssh
```



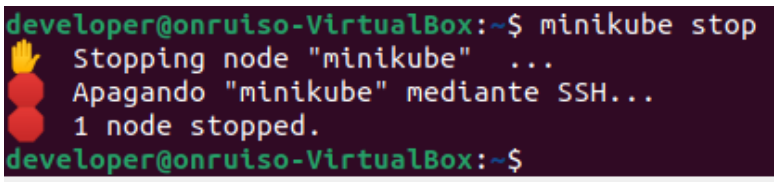
Para salir del SHELL.

```
exit
```



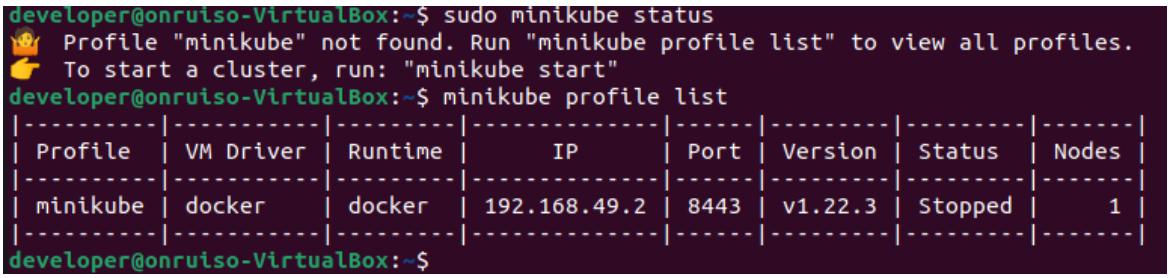
Para terminar de ejecutar un cluster de un solo nodo.

```
minikube stop
```



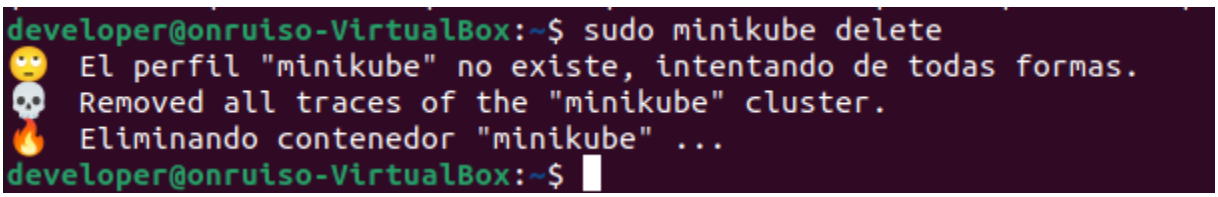
Comprobar el estado de Minikube.

```
minikube status
```



Para eliminar un cluster de un solo nodo.

```
minikube delete
```



Para ver una lista de complementos de Minikube instalados.

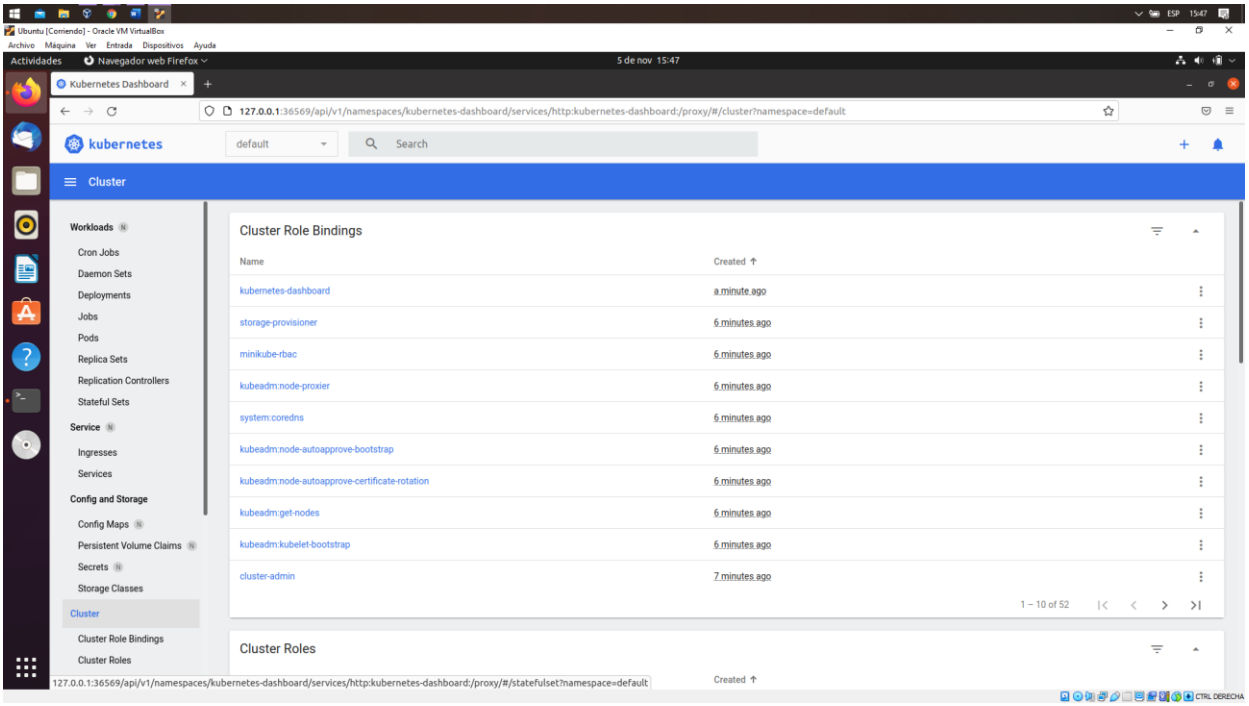
```
minikube addons list
```

```
developer@onruiso-VirtualBox:~$ sudo minikube addons list
|-----------------------------|-----------------------|
|         ADDON NAME          |      MAINTAINER       |
|-----------------------------|-----------------------|
| ambassador                  | unknown (third-party) |
| auto-pause                  | google                |
| csi-hostpath-driver         | kubernetes            |
| dashboard                   | kubernetes            |
| default-storageclass        | kubernetes            |
| efk                         | unknown (third-party) |
| freshpod                    | google                |
| gcp-auth                    | google                |
| gvisor                      | google                |
| helm-tiller                 | unknown (third-party) |
| ingress                     | unknown (third-party) |
| ingress-dns                 | unknown (third-party) |
| istio                       | unknown (third-party) |
| istio-provisioner           | unknown (third-party) |
| kubevirt                    | unknown (third-party) |
| logviewer                   | google                |
| metallb                     | unknown (third-party) |
| metrics-server              | kubernetes            |
| nvidia-driver-installer     | google                |
| nvidia-gpu-device-plugin    | unknown (third-party) |
| olm                         | unknown (third-party) |
| pod-security-policy         | unknown (third-party) |
| portainer                   | portainer.io          |
| registry                    | google                |
| registry-aliases            | unknown (third-party) |
| registry-creds              | unknown (third-party) |
| storage-provisioner         | kubernetes            |
| storage-provisioner-gluster | unknown (third-party) |
| volumesnapshots             | kubernetes            |
|-----------------------------|-----------------------|
developer@onruiso-VirtualBox:~$
```

Acceder al panel de Minikube, este es un complemento de panel predeterminado de la dependencia. El panel es web y proporciona una forma de administrar un CLUSTER de Kubernetes sin ejecutar comandos en la terminal.

```
minikube dashboard
```

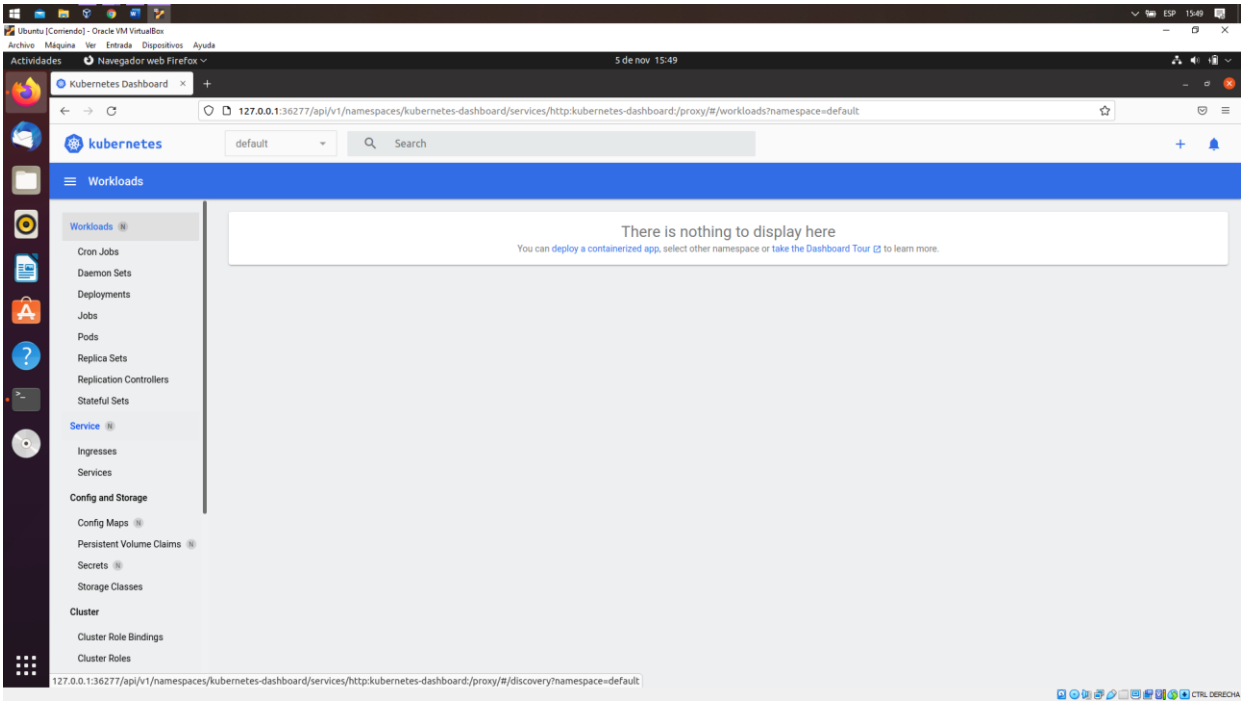
```
developer@onruiso-VirtualBox:~$ minikube dashboard
🤔  Habilitando dashboard
    ▪ Using image kubernetesui/metrics-scraper:v1.0.7
    ▪ Using image kubernetesui/dashboard:v2.3.1
🤔  Verifying dashboard health ...
🚀  Launching proxy ...
🤔  Verifying proxy health ...
🎉  Opening http://127.0.0.1:36569/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
```



Una vez se salga del terminal, el proceso se terminará y el panel de Minikube se apagará. De forma alternativa se puede acceder al panel de control directamente a través de un navegador de internet. Para poder acceder a este panel se puede adquirir la dirección IP con el siguiente comando.

```
minikube dashboard --url
```

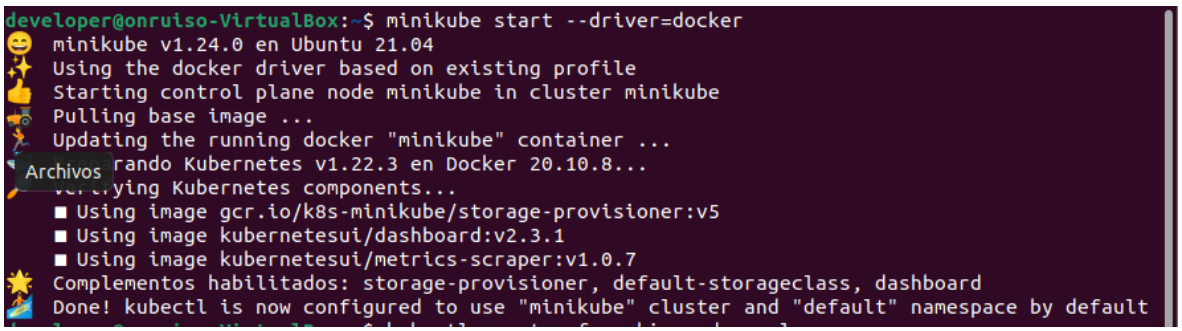
```
developer@onruiso-VirtualBox:~$ minikube dashboard --url
🤔  Verifying dashboard health ...
🚀  Launching proxy ...
🤔  Verifying proxy health ...
http://127.0.0.1:36277/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/
```



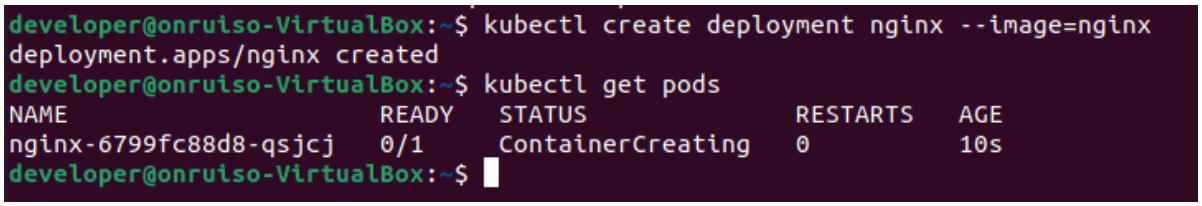
EJEMPLO DE USO DE KUBERNETES

Iniciamos la dependencia de minikube.

Minikube start –driver=Docker

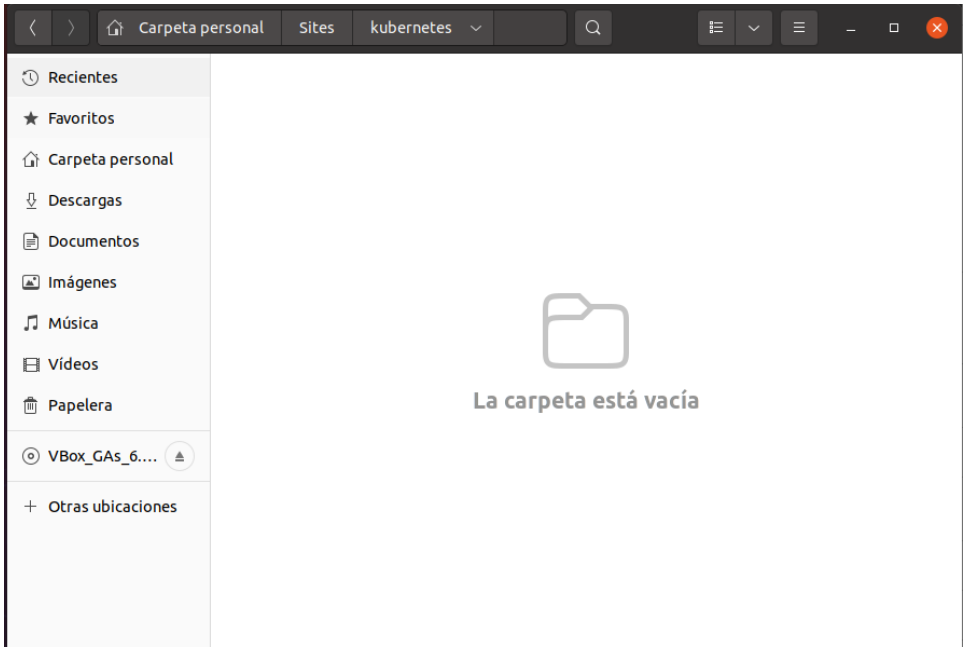


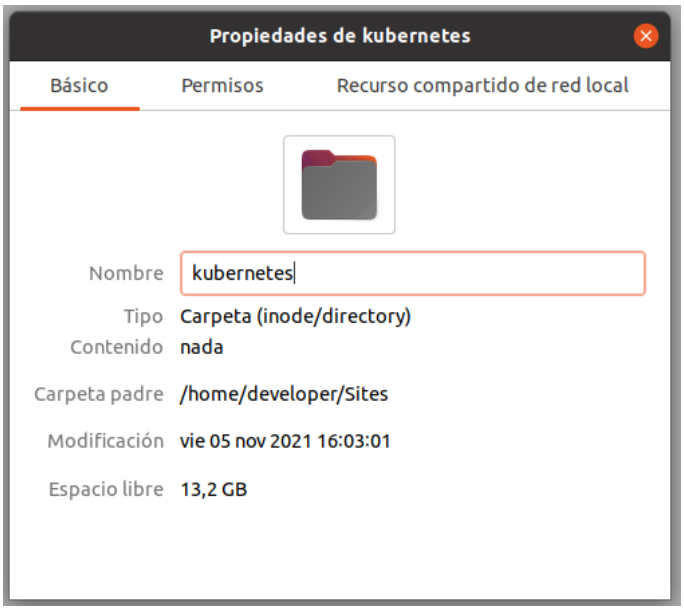
kubectl create deployment nginx --image=nginx
kubectl get pods



Creamos la ruta y el archivo que lanzaremos.

/home/developer/Sites

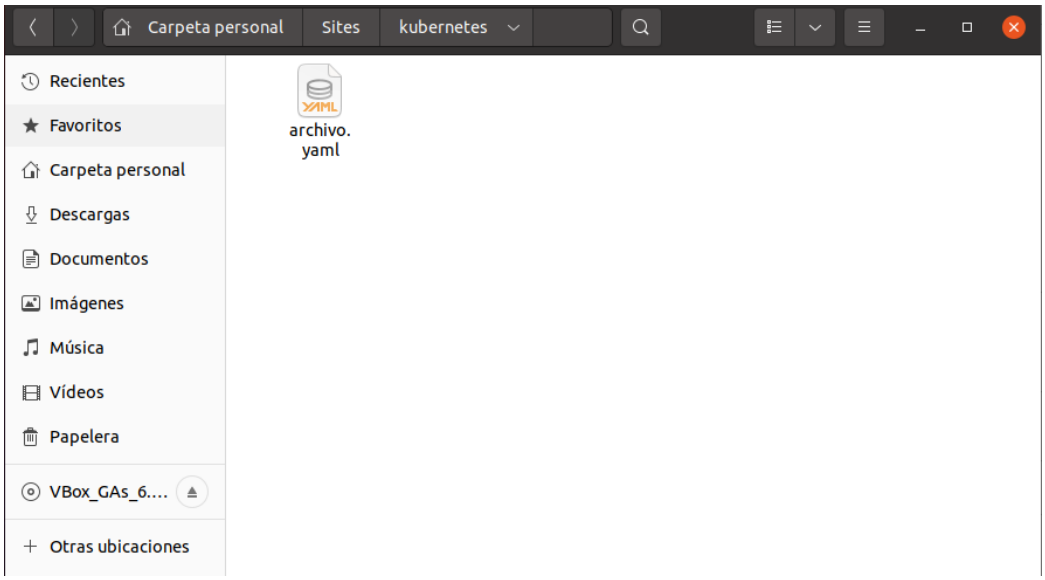




touch archivo.yaml

```
developer@onruiso-VirtualBox:~/Sites/kubernetes$ touch archivo.yaml
developer@onruiso-VirtualBox:~/Sites/kubernetes$
```

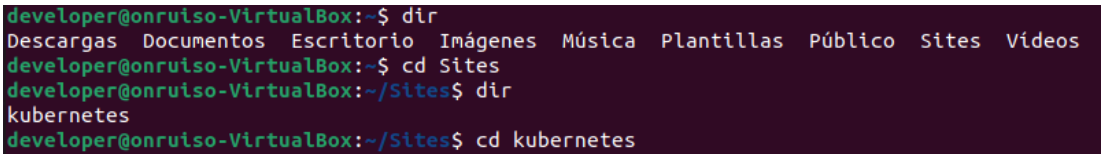
Abrimos el archivo con un editor de texto.




```
apiVersion: v1
kind: Pod
metadata:
  name: my-nginx
  # Especificamos que el pod tenga un label con clave "app" y valor "nginx"que será lo que vea el RC para saber que tiene que gestionarlo.
  labels:
```

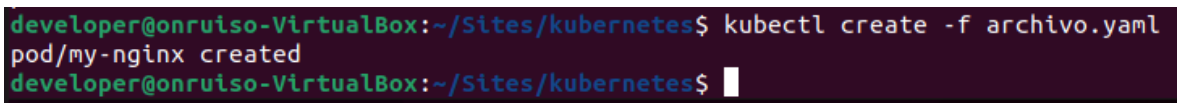
```
    app: nginx
spec:
  containers:
    - name: apache
      image: apache
      ports:
        - containerPort: 80
  restartPolicy: Always
```

Accedemos a la ruta por terminal



Levantamos el servicio.

```
kubectl create -f archivo.yaml
```



Confirmamos que se haya creado el POD