INSTITUTO DE TECNOLOGIAS SUPERIOR SUDAMERICANO

TENDENCIAS TECNOLÓGICAS

Nathaly Pinguil

Cuarto Ciclo

Informe practica #1

Ing. Marco Guaman

14/10/2024

I. Título
Contenedores nginx

II. Tiempo de duración
20 minutos

III. Fundamentos:
Docker simplifica la creación y ejecución de aplicaciones al empaquetarlas en contenedores, que son fácilmente transportables y ejecutables en cualquier entorno.

Un contenedor es como una caja virtual que contiene una aplicación y todo lo necesario para su funcionamiento, lo que facilita su transporte y ejecución en cualquier lugar.

El comando docker run se utiliza para crear y ejecutar un contenedor a partir de una imagen de Docker. Descarga la imagen del registro de Docker Hub si no está presente localmente, y luego crea y ejecuta un contenedor basado en esa imagen.

El argumento --name permite especificar un nombre personalizado para el contenedor que se está creando. En el ejemplo proporcionado, el nombre del contenedor se establece como "nginx".

El argumento -d indica a Docker que ejecute el contenedor en modo "detached" o desvinculado. Esto significa que el contenedor se ejecutará en segundo plano, permitiendo continuar utilizando la terminal sin que el contenedor ocupe el control.

El argumento -p se utiliza para mapear los puertos del host a los puertos del contenedor. En el ejemplo, se está mapeando el puerto 8089 del host al puerto 80 del contenedor. Esto redirige cualquier tráfico que llegue al puerto 8089 del host al puerto 80 dentro del contenedor.

"nginx" es el nombre de la imagen de Docker que se utilizará para crear el contenedor. En este caso, se está utilizando la imagen "nginx", que es un servidor web ligero y de alta eficiencia.

En resumen, el comando docker run --name nginx -d -p 8089:80 nginx crea un contenedor con el nombre "nginx", utilizando la imagen de Docker "nginx", ejecutándose en segundo plano y mapeando el puerto 8089 del host al puerto 80 del contenedor.

IV. Conocimientos previos.

Referencia de Comandos en Linux

Traducción: Alex Barrios (alexbariv@gmail.com) / Basado en la publicación original de Jacob en FOSSWire

Manejo de Archivos

ls - listar directorio
ls -al - listado con formato y mostrando ocultos
cd *dir* - cambiar a directorio "dir"
cd - cambiar a directorio home
pwd - muestra el directorio actual
mkdir *dir* - crear el directorio "dir"
rm *archivo* - borrar archivo
rm -r *dir* - borrar directorio "dir"
rm -f *archivo* - forzar el borrar archivo
rm -rf *dir* - forzar borrar directorio de forma recursiva
cp *archivo1 archivo2* - copiar *archivo1 a archivo2*
cp -r *dir1 dir2* - copiar *dir1 a dir2*; Creando *dir2 si no existe*
mv *archivo1 archivo2* - renombrar o mover *archivo1* a *archivo2*. Si el *archivo2* es un directorio, mueve *archivo1* al contenido de ese directorio
ln -s *archivo link* - crea un enlace simbólico de link a *archivo*
touch archivo - crea o actualiza un archivo
cat > archivo - coloca la salida estándar en *archivo*
more archivo - muestra el contenido de un archivo
head archivo - muestra las primeras 10 líneas de un archivo
tail archivo - muestra las últimas 10 líneas de un archivo
tail -f *file* - muestra las últimas 10 líneas de en tiempo real

Gestión de procesos

ps - muestra los procesos activos actualmente
top - muestra todos los procesos
kill *pid* - mata un proceso indicando el *pid*
killall *proc* - mata un proceso llamado *proc*
bg - lista los procesos detenidos o trabajando en fondo; puede resumir procesos
fg - trae el proceso más reciente al frente
fg *n* - trae N procesos al frente

Permisología

chmod *octal archivo* - cambia los permisos del *archivo* con *octal*, que pueden ser identificados por separado el usuario, grupo o mundo añadiendo:
* 4 - leer (r)
* 2 - escribir (w)
* 1 - ejecutar (x)

Ejemplos:
chmod 777 - leer, escribir, y ejecutar para todos
chmod 755 - rwx para el dueño, rx para el grupo y mundo
Para más opciones, observa: **man chmod**.

Uso de SSH

ssh *usuario@host* - conecta a *usuario* en *host*
ssh -p *puerto user@host* - conecta a el *host* en el *puerto* con el usuario *user*
ssh-copy-id *user@host* - añade tu llave a el *host* para activar el inicio de sesión sin clave

Búsquedas

grep *patrón archivos* - busca en los archivos por el *patrón*
grep -r patrón *dir* - busca recursivamente el *patrón* en los directorios.
comando | grep *patrón* - busca por el *patrón en la salida del comando*
locate *archivo* - encuentra todas las instancias del archivo

Información del Sistema

date - muestra la hora y fecha actual
cal - muestra el calendario del mes
uptime - muestra el tiempo en ejecución del sistema
w - muestra quién está conectado
whoami - muestra como quién está conectado
uname -a - muestra información del kernel
cat /proc/cpuinfo - información del cpu
cat /proc/meminfo - información de la memoria
man *comando* - muestra el manual para el *comando*
df - muestra el uso de disco
du - muestra el uso de disco del directorio
free - muestra la memoria ram y swap libre
whereis app - muestra las posibles ubicaciones de *app*
which *app* - muestra cual *app* corre por defecto

Compresión

tar cf *file.tar archivos* - crear un archivo tar llamado *file.tar* que contiene *archivos*
tar xf *file.tar* - extrae los contenidos de *file.tar*
tar czf *file.tar.gz files* - crea un tar comprimido con Gzip
tar xzf *file.tar.gz* - extrae un tar que usa Gzip
tar cjf *file.tar.bz2* - crea un tar comprimido con Bzip2
tar xjf *file.tar.bz2* - extrae un tar que usa Bzip2
gzip *file* - comprime *file y lo renombra a file.gz*
gzip -d *file.gz* - descomprime *file.gz* de vuelta a *file*

Redes

Ping *host* - ejecuta ping a *host y muestra los resultados*
whois *dominio* - obtiene la info whois de un dominio
dig *dominio* - obtiene la info DNS de un dominio
dig -x *host* - busca el reverso DNS del *host*
wget archivo - descarga un archivo
wget -c archivo - continua una descarga pausada

Instalando Software

Instalando desde las fuentes (normalmente un tar.gz):
./configure
make
make install
Con sistemas de paquetes
dpkg -i *pkg.deb* - instala un paquete (Debian)
rpm -Uvh *pkg.rpm* - instala un paquete (RPM)

Atajos de teclado

Ctrl+C - detiene el comando actual
Ctrl+Z - pausa el comando actual, lo resumes con **fg** al frente o **bg** en el fondo
Ctrl+D - sierra la sesión, similar a **exit**
Ctrl+W - borra una palabra de la linea actual
Ctrl+U - borra toda la línea
Ctrl+R - repite el último comando
exit - sale de la sesión actual

Dominar los comandos de Vi es esencial para mejorar la fluidez y la eficiencia en la práctica, lo que permite optimizar el tiempo dedicado a la edición de archivos. Además, tener un buen dominio de estos comandos es fundamental, ya que el docente ha destacado su importancia para la programación con Vi.

Abrir un archivo con vi: Para abrir un archivo en Vi, simplemente escribe vi nombre_del_archivo en la terminal y presiona Enter.

Modo de edición: Una vez dentro del archivo, presiona la tecla i para ingresar al modo de edición, donde puedes insertar texto o realizar ediciones.

Salir del modo de edición: Para salir del modo de edición y volver al modo de comando, simplemente presiona la tecla Esc.

Guardar cambios y salir: Si estás en el modo de comando, presiona : para entrar en el modo de comandos, luego escribe wq y presiona Enter para guardar los cambios y salir de Vi.

Salir sin guardar cambios: Si prefieres salir de Vi sin guardar los cambios, estando en el modo de comando, presiona : para entrar en el modo de comandos, luego escribe q! y presiona Enter.

Crear un nuevo archivo con Vi: Para crear un nuevo archivo con Vi, simplemente abre Vi con un nombre de archivo que no exista, por ejemplo: vi nuevo_archivo.txt.

V. Objetivos a alcanzar

- Implementar contenedores con nginx utilizando Docker.
- Comprender el funcionamiento de los contenedores con nginx, después de implementar el contenedor, se podrá explorar su funcionamiento accediendo al servidor web nginx desde mi navegador web.
- Manipular archivos de configuración de nginx,utiliza archivos de configuración para definir su comportamiento, se puede manipular estos archivos para personalizar la configuración del servidor web según nuestros requisitos.

VI. Equipo necesario:

- Computador con sistema operativo Windows/Linux.
- Play with Docker
- Comandos de Linux
- Internet

VII. Material de apoyo.

- Documentación de docker.
- Guía de asignatura
- El docente
- Comando

VIII. Procedimiento

Iniciamos accediendo al sitio web de Play with Docker, donde nos registramos y comenzamos a trabajar en la consola, ejecutando comandos de Linux para comenzar nuestra práctica.

Después, nos elevamos como administradores ejecutando sudo su, lo que nos permite tener privilegios administrativos. Luego, utilizamos docker ps para visualizar nuestro primer

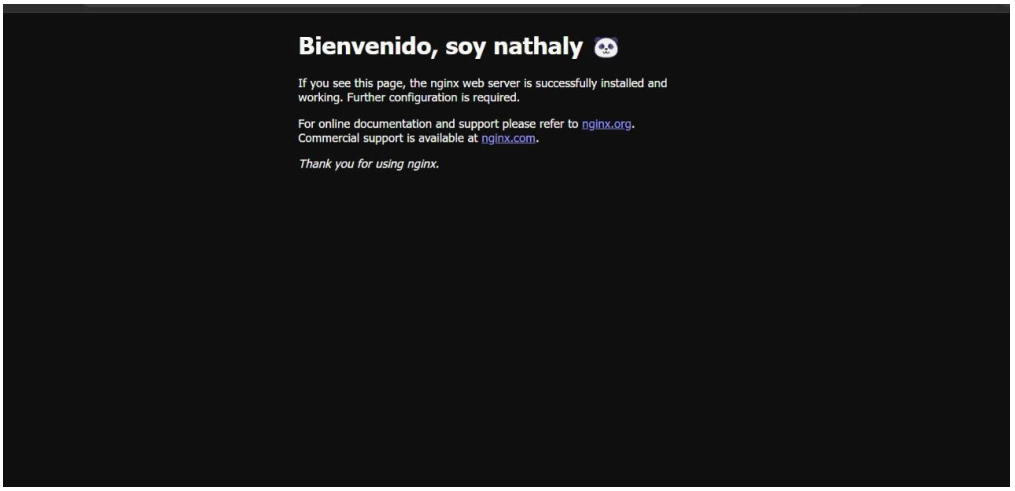
contenedor en ejecución. Posteriormente, ejecutamos el comando docker run --name nginx -d -p 8089:80 nginx para descargar y ejecutar un contenedor de nginx, con el objetivo de iniciar nuestro proyecto.

Una vez dentro del contenedor, aplicamos nuestras habilidades en Linux utilizando el editor de texto Vi. El propósito principal en esta sesión era modificar los archivos dentro de los contenedores. Como ejemplo, se nos presentó un archivo HTML y comenzamos a ejecutar los comandos necesarios para extraer y editar su contenido utilizando Vi.

Luego de realizar las modificaciones deseadas, guardamos los cambios y salimos del editor Vi. Específicamente, accedemos al nombre del archivo dentro del contenedor para realizar esta acción, siguiendo el ejemplo proporcionado.

Por último, cargamos todos los cambios realizados y confirmamos que se han aplicado correctamente. Al ver una ventana emergente indicando el éxito de nuestras modificaciones, podemos estar seguros de que el proceso ha sido completado con éxito.

IX. Resultados esperados:



Se puede ya visualiza los cambios

X. Bibliografía

Breus, V. (2023, April 18). *Como correr Nginx en un parche de Docker Contenedor en Ubuntu - manual de usuario*. Serverspace.io; ITGLOBAL.COM NL. https://serverspace.io/es/support/help/how-to-run-nginx-docker-container-ubuntu

Docker. (n.d.). Docker.com. Retrieved April 16, 2024, from https://hub.docker.com/_/nginx

Landajuela, I. (n.d.). *Montar un servidor Web básico con NGINX y Docker*. Blog personal. Retrieved April 16, 2024, from https://soka.gitlab.io/blog/post/2019-07-08-docker-imagenes-y-contenedores/