Network Administration/System Administration (NTU CSIE, Spring 2024) Homework #5 - Virsh, Docker & Kubernetes

B12902110 呂承諺

March 29, 2024

Virsh

1. **Command**

```
virt-install \
  --name b12902110 \
  --vcpus 8 \
  --memory 8192 \
  --disk /tmp2/b12902110/nasahw5/ubuntu.qcow2,format=qcow2,size=20 \
  --network user,mac=52:54:00:90:21:10 \
  --graphics type=vnc,password=902110,port=5950,listen=0.0.0.0 \
  --cdrom /tmp2/nasa-hw5/ubuntu.iso
```

Result

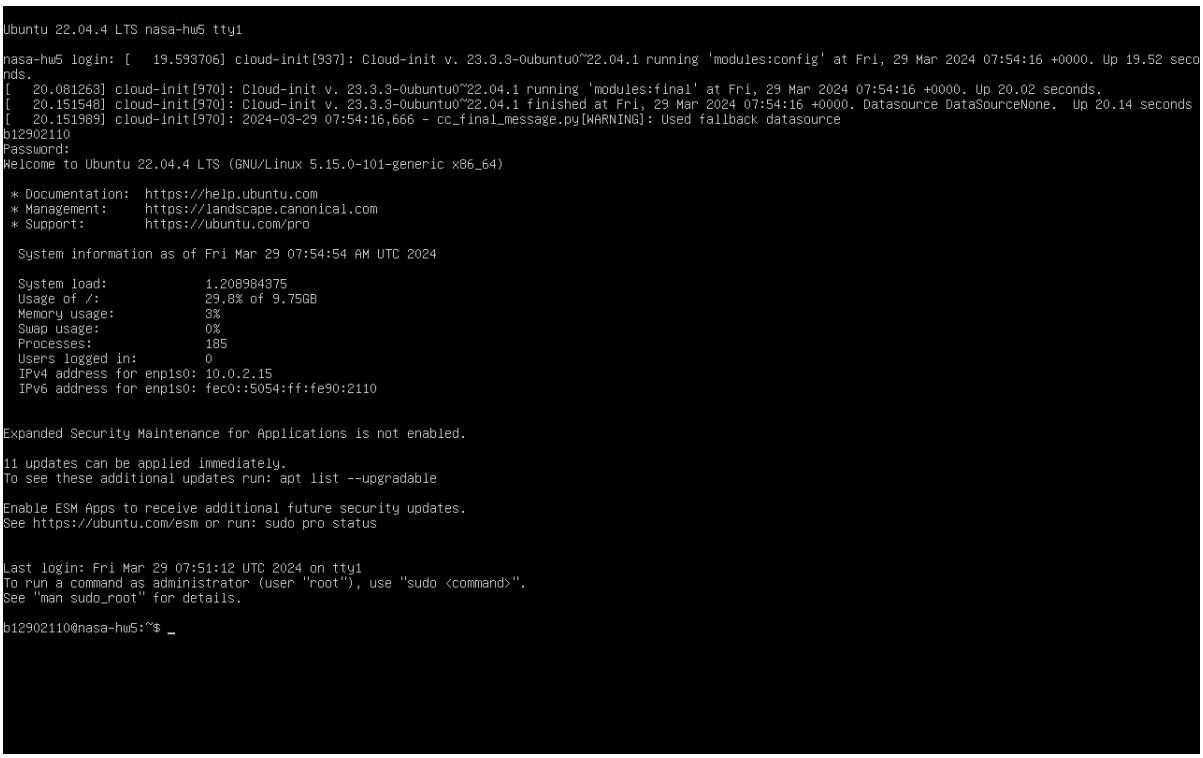
`virsh list`:

```
b12902110@ws1: ~
$ virsh list
 Id   Name         State
--------------------------
 4    b12902110    running
```

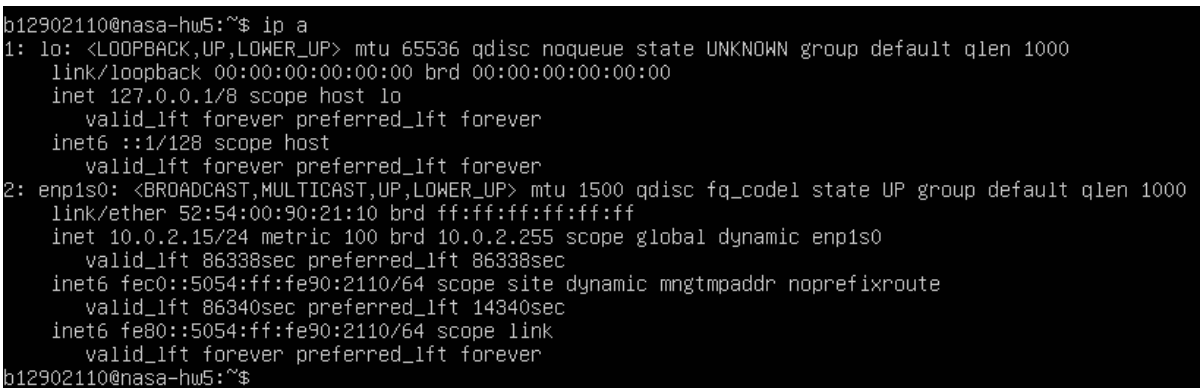
After boot up:

```
Ubuntu 22.04.4 LTS nasa-hw5 tty1

nasa-hw5 login: [   19.593706] cloud-init[937]: Cloud-init v. 23.3.3-0ubuntu0~22.04.1 running 'modules:config' at Fri, 29 Mar 2024 07:54:16 +0000. Up 19.52 seco
nds.
[   20.081263] cloud-init[970]: Cloud-init v. 23.3.3-0ubuntu0~22.04.1 running 'modules:final' at Fri, 29 Mar 2024 07:54:16 +0000. Up 20.02 seconds.
[   20.151548] cloud-init[970]: Cloud-init v. 23.3.3-0ubuntu0~22.04.1 finished at Fri, 29 Mar 2024 07:54:16 +0000. Datasource DataSourceNone.  Up 20.14 seconds
[   20.151989] cloud-init[970]: 2024-03-29 07:54:16,666 - cc_final_message.py[WARNING]: Used fallback datasource
```

After login:



`ip a`:



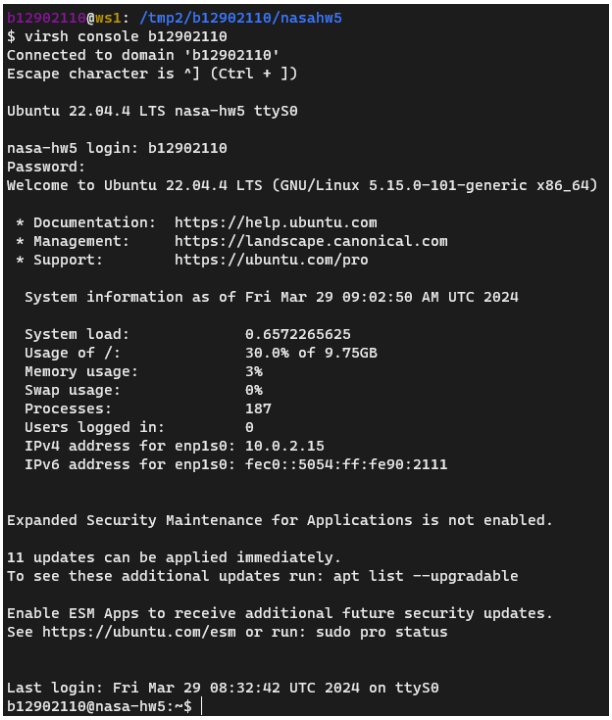
2. **Command (on VM)**

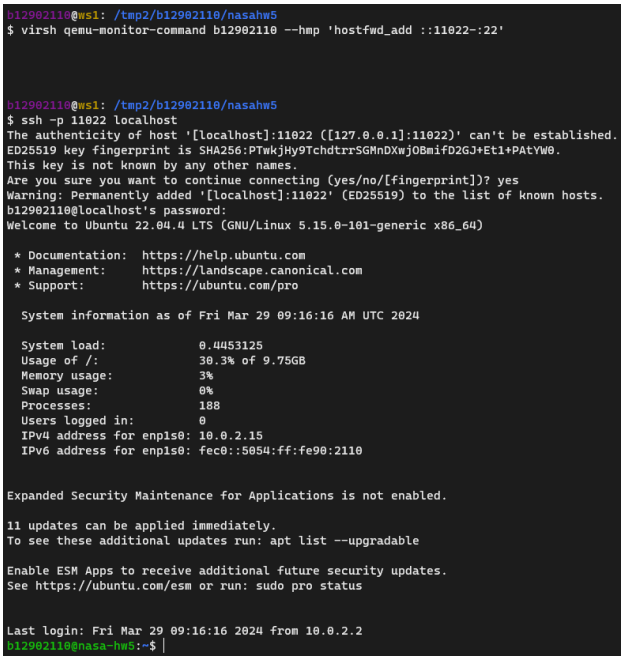
```
sudo systemctl enable --now serial-getty@ttyS0.service
sudo systemctl enable --now ssh.service
```

Command (on host)

```
virsh qemu-monitor-command b12902110 --hmp 'hostfwd_add ::11022-:22'
```

Result





References

- `man virt-install`, `man virsh`
- libvirt: Network XML format
- libvirt: Creating a NAT Virtual Network
- libvirt: NAT forwarding (aka "virtual networks")
- libvirt: Domain XML format
- qemu - How to port forward SSH in virt-manager? - Unix & Linux Stack Exchange
- Invocation —QEMU documentation
- host port forward with qemu through libvirt in user-mode networking - Server Fault
- Structured Procrastination port redirection from kvm host to guest –Structured Procrastination

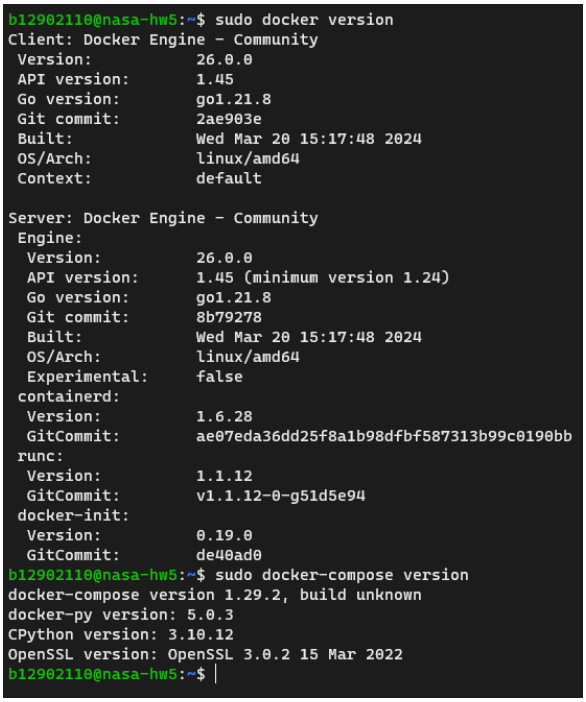
Docker

Set Up

3. **Command**

```
# Add Docker's official GPG key:
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg \
  -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] \
  https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update

# Install the Docker packages.
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin \
  docker-compose-plugin docker-compose
```

Result



Docker Basics

4. Cases to use Docker:

   - Host OS is Linux.
   - Application runs on Linux.
   - When performance and efficiency is important, because Docker shares resources with the host.
   - When we want to deploy the environment elsewhere, Docker offers better portability.

Cases to use VM:

- Host OS is not Linux.
- Application doesn't run on Linux.
- Need control over hardware resources (CPU, memory, etc.).

References

- Docker vs VM - Difference Between Application Deployment Technologies - AWS
- Reddit - Dive into anything

5. The Docker Engine runs on a Linux kernel. On Windows and macOS, Docker is run in a virtual machine (WSL2 or LinuxKit VM), which hurts performance.

   References

   - Instantly Improve Docker Performance on Mac
   - Docker on MacOS is slow and how to fix it | CNCF
   - Install Docker Desktop on Windows | Docker Docs

6. (a) `docker stop $(docker ps -a -q)`

   "`docker ps -a -q`" list IDs of all containers. Substitute it into argument of "`docker stop`".

   (b) `docker image rm $(docker image -a -q)`

   "`docker image -a -q`" list IDs of all images. Substitute it into argument of "`docker image rm`".

   (c) `docker system prune`

   Remove all unused containers, networks, images (both dangling and unused), and optionally, volumes.

   (d)
   ```
   docker inspect \
     --format='{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' \
     5b0f1ed0dcb8
   ```

   Return low-level information on Docker objects.

   (e) `docker container stats`

   Display a live stream of container(s) resource usage statistics.

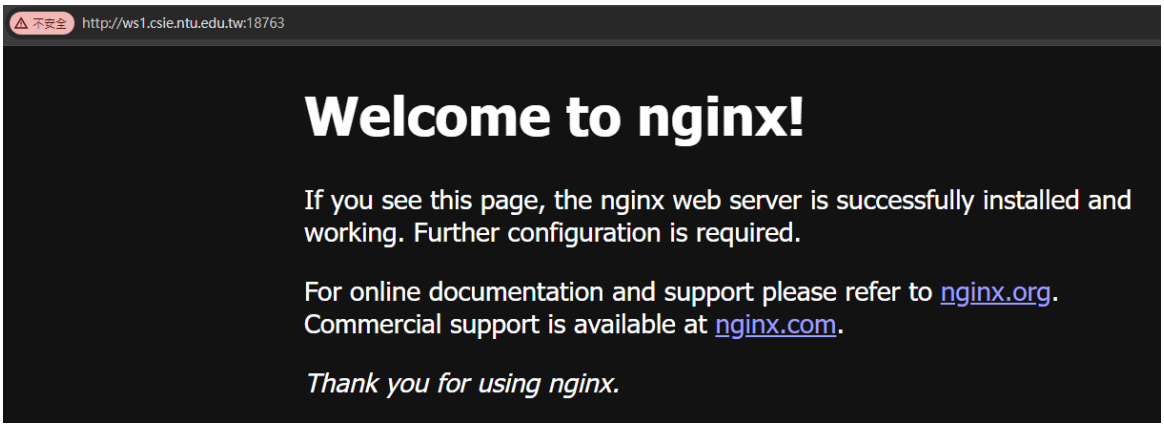
   References

   - Stop and remove all docker containers - Stack Overflow
   - docker system prune | Docker Docs
   - How to get a Docker container's IP address from the host - Stack Overflow
   - docker inspect | Docker Docs
   - docker container stats | Docker Docs

7. **Command** `docker run --name nginx-1 -d -p 8763:80 nginx:1.24.0`

   - `docker run`: Create and run a new container from an image.
   - `--name nginx-1`: Assign name `nginx-1` to container.
   - `-d`: Run in background.
   - `-p 8763 80`: Forward port 8763 on host to port 80 in container.
   - `nginx:1.24.0`: Image to run.

   Result



   Note: Port 18763 on ws1 is forwarded to port 8763 on Ubuntu VM.

   References

   - `docker run --help`

8. **Command** `docker exec -it nginx-1 /bin/bash`

   - `docker exec`: Execute a command in a running container.
   - `-i`: Keep STDIN open even if not attached.
   - `-t`: Allocate a pseudo-TTY.
   - `nginx-1`: Container name.
   - `/bin/bash`: Command to execute.

   Result

```
b12902110@nasa-hw5:~/nasa-hw5/b12902110$ docker exec -it nginx-1 /bin/bash
root@4284f880e2a8:/#
```

   References

   - `docker exec --help`

9. **Command** `docker exec nginx-1 cat /etc/nginx/nginx.conf`

   Result

```
b12902110@nasa-hw5:~/nasa-hw5/b12902110$ docker exec nginx-1 cat /etc/nginx/nginx.conf

user  nginx;
worker_processes  auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;


events {
    worker_connections  1024;
}


http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log  /var/log/nginx/access.log  main;

    sendfile        on;
    #tcp_nopush     on;

    keepalive_timeout  65;

    #gzip  on;

    include /etc/nginx/conf.d/*.conf;
}
```

10. (a) `ENTRYPOINT` defines the executable that the container runs, `CMD` defines default arguments for `ENTRYPOINT`.

    (b) `ENTRYPOINT` is overridden with `docker run --entrypoint NEW_ENTRY_POINT ...`. `CMD` is overridden with `docker run ... NEW_COMMAND`.

    (c) If using exec form, `CMD` is the arguments for `ENTRYPOINT`.

    References

    - Dockerfile reference | Docker Docs

11. Docker Compose is a tool to manage *multi-container* applications. We define services, network, volumes to be used in an YAML configuration file. On the other hand, Docker is the underlying virtualization engine that handles containers, images, networks, etc.

    References

    - Docker Engine overview | Docker Docs
    - Docker Compose overview | Docker Docs

12. (a) **Command** `docker-compose up -d`

    - `docker-compose up`: Builds, (re)creates, starts, and attaches to containers for a service.
    - `-d`: Detached mode: Run containers in the background.

    (b) **Command** `docker-compose pause`

    Pause services.

    (c) **Command** `docker-compose down -v`

    - `docker-compose down`: Stops containers and removes containers, networks, volumes, and images created by ‘up‘.

- `-v`: Remove volumes.

References

- `docker-compose --help`
- `docker-compose up --help`
- `docker-compose pause --help`
- `docker-compose down --help`

Application

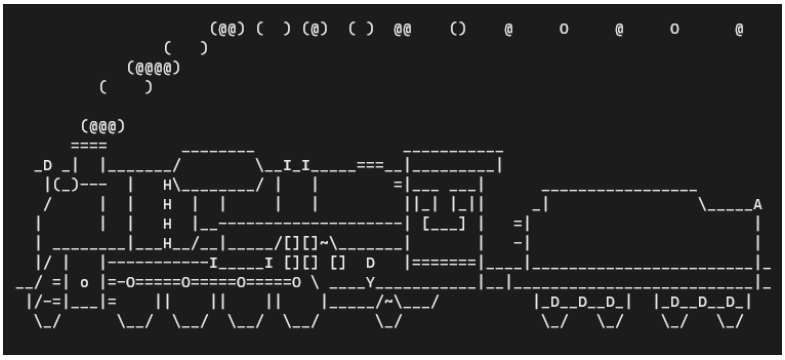
13. **b12902110.Dockerfile**

```
FROM alpine:latest
RUN apk update && apk add build-base git ncurses-dev
RUN git clone https://github.com/mtoyoda/sl.git
WORKDIR /sl
RUN make
CMD [ "/sl/sl" ]
```

Command

```
sudo docker build -t sl - < b12902110.Dockerfile
sudo docker run -it --rm sl
```

Result



References

- alpine - Official Image | Docker Hub
- Alpine Package Keeper - Alpine Linux
- GCC - Alpine Linux
- Alpine Linux packages - build-base
- Alpine Linux packages - git
- Alpine Linux packages - ncurses-dev
- Dockerfile reference | Docker Docs
- docker build | Docker Docs

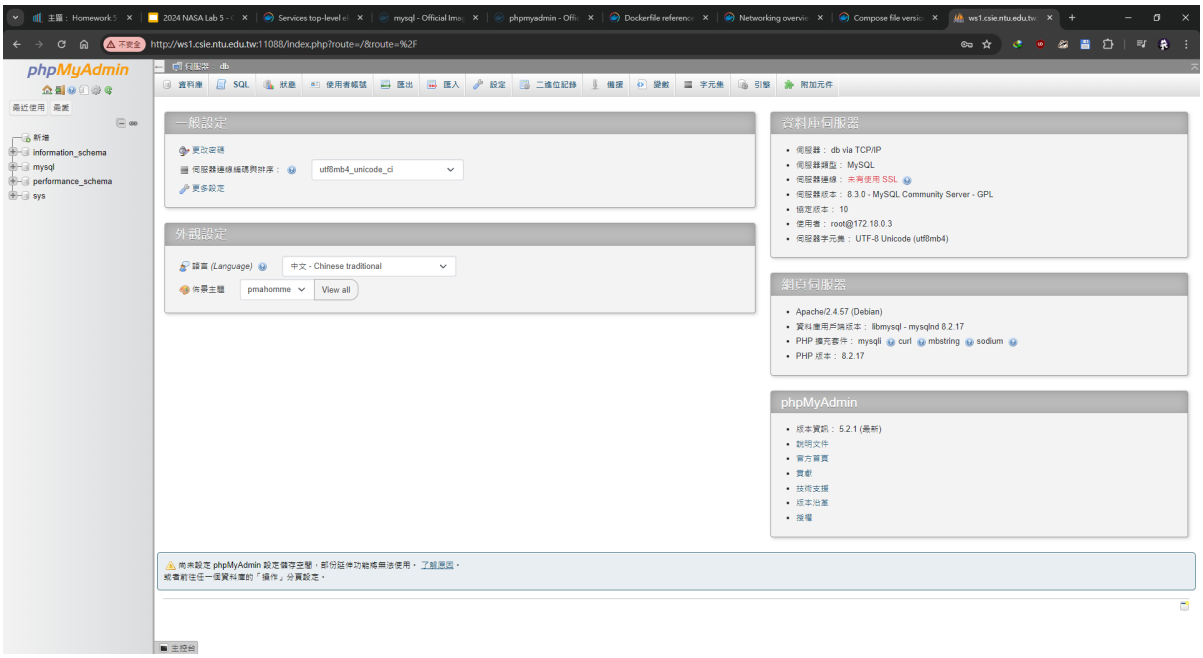
14. **mysql-b12902110.yaml**

```yaml
services:
  db:
    image: mysql:latest
    networks:
      - nasa-net
    environment:
      - MYSQL_HOST=b12902110
      - MYSQL_ROOT_PASSWORD=secret

  web:
    image: phpmyadmin:latest
    depends_on:
      - db
    ports:
      - 8888:80
    networks:
      - nasa-net

networks:
  nasa-net:
```

Command

```
sudo docker-compose -f mysql-b12902110.yaml up
```

Result



References

- Try Docker Compose | Docker Docs
- Networking in Compose | Docker Docs
- mysql - Official Image | Docker Hub
- phpmyadmin - Official Image | Docker Hub

- MySQL :: MySQL 5.7 Reference Manual :: 4.9 Environment Variables
- Ways to set environment variables with Compose | Docker Docs

Kubernetes

15.

16. (a) **Steps**

    i. Reuse the VM installed in step 1.

    ii. Run the following commands:

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
alias kubectl="minikube kubectl --"
kubectl
```

(b) **nginx-b12902110.yaml**

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-b12902110
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx-b12902110
  template:
    metadata:
      labels:
        app: nginx-b12902110
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 8888
```

mysql-b12902110.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-b12902110
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql-b12902110
  template:
    metadata:
      labels:
        app: mysql-b12902110
    spec:
      containers:
      - name: mysql
        image: mysql:latest
        env:
```

```
        - name: MYSQL_HOST
          value: b12902110
        - name: MYSQL_ROOT_PASSWORD
          value: $(MYSQL_ROOT_PASSWORD)
```

phpmyadmin-b12902110.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: phpmyadmin-b12902110
spec:
  replicas: 1
  selector:
    matchLabels:
      app: phpmyadmin-b12902110
  template:
    metadata:
      labels:
        app: phpmyadmin-b12902110
    spec:
      containers:
      - name: phpmyadmin
        image: phpmyadmin:latest
```

(c)