Docker Case Study - Automate Infrastructure allocation for Learning & Development.

Swastik Shrivastava

IMT2016055

Requirements:-

→You need a standard Linux Docker image to create a container.
→Each user should have independent Linus System.
→Specific Environment is to be created in the container as desired.
→User should not get access to other container or even the docker command.
→Admin can monitor participant containers.
→Automate container creation and deletion for ease.

Creating and allocating containers to users:

1. To create the container from the selected image, execute the command:

$docker create -i -t --name<container name><docker image>

/bin/bash

(the -i option keeps standard input open, -t allocates pseudo-terminal for conatiner, and --name assigns a name for the container).

2. Start and Attach the container:
Use following command lines:

$docker start <container name>
$docker attach <container name>

3.) Install the required applications using following commands:

apt update
apt install vim
apt install gcc

4.) Commit the container:

```
$ docker commit -a "name" ee51c71aa6fa <container name image>
```

5.)We wrote a shell script “create_container.sh” to *automatically allocate* resources(container) to every user.

- Users.txt

- User1
  User 2
  User 3

- createContainers.sh
- Echo -n “Enter name to create a user file:

```
read file
while read user
    do
        Docker create -it --name $user <docker image>/bin/bash
    Done <$file
```

6.) Run the shell script “create_container.sh” corresponding to the users text file to create a separate container for each user defined in the text file.

7.) we can write another shell script to attach multiple images to our container using another file let say- “attach_container.sh”

➨attach_container.sh

```
echo -n “Enter the name : ”
read name
        docker start $name
        docker attach $name
```

Monitoring the containers:

To monitor the resources used by each containers we can execute the following command:

```
$docker stats
$(docker ps -a| grep user | awk ‘{print $NF}’)
```

➜The docker stats command with container id or name will print the computing resource utilization.

➜The (docker ps -a) will print the list of all running container detains for all the users.

➜The grep user will help filter with only the user’s container name.

➜To extract the container name of all users, cut the line with only the last field (awk '{print $NF}')

Container Deletion:

To automate the container deletion we can again write a script to make the work easier:

delete_container.sh

```
Echo -n “Enter user list :”
read file
While read user
do
        docker stop $user
        docker rm $user
done <$file
```

Now you can can directly run this delete container shell file to automate the deletion.

NOTE :

To ensure that a user remain in the boundary of his container and cannot interfere with other containers.

Add the following lines to every user's .bashrc

`docker start -ai <name>`
`exit`