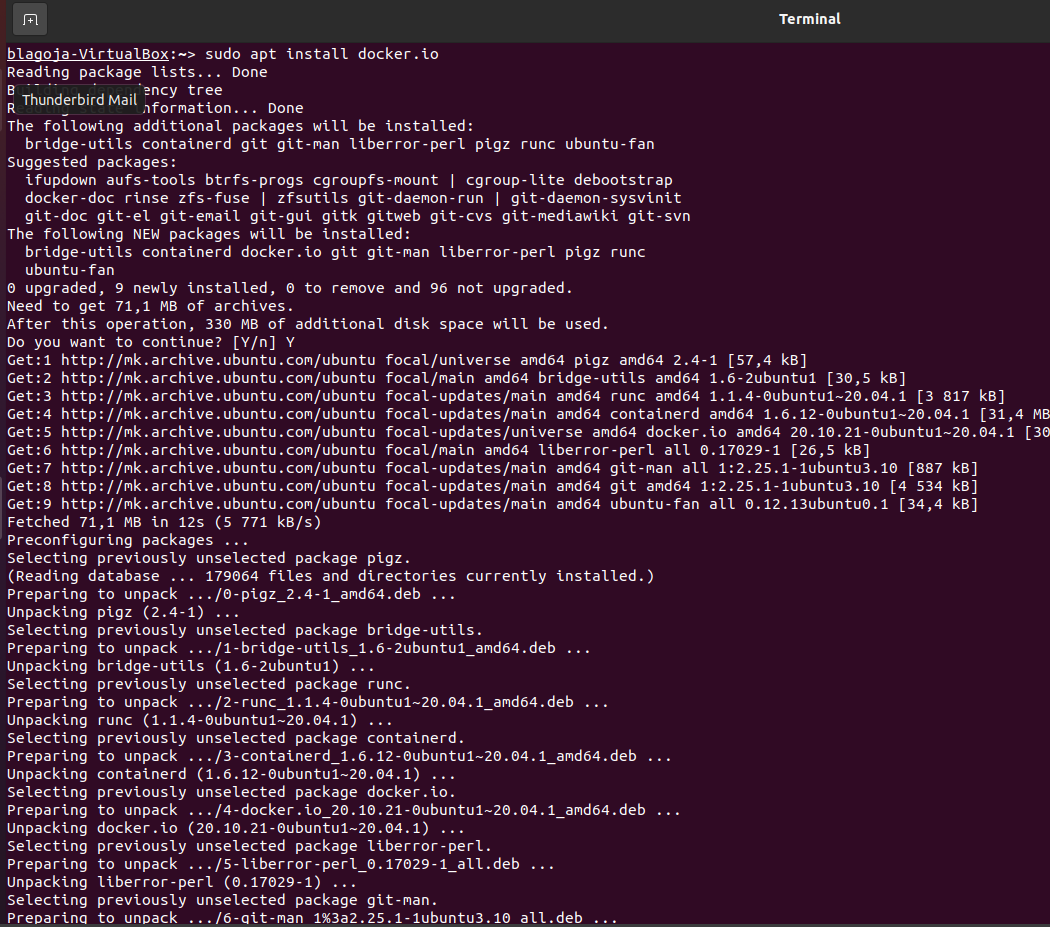
1. Log in to your VM.

2. Start terminal and elevate your privileges to root.

3. Run yum install docker.

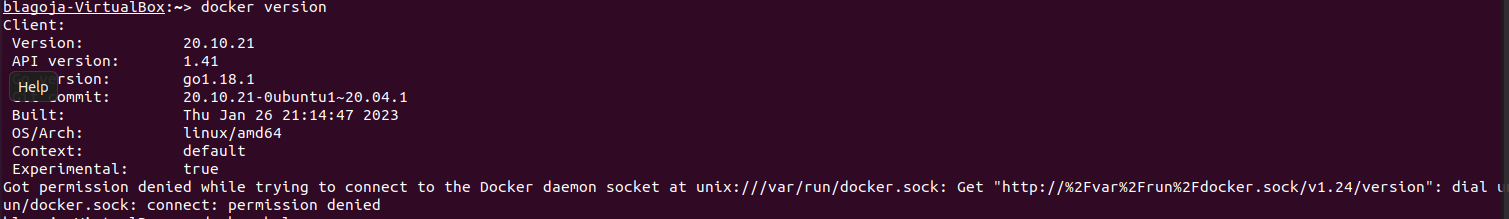


4. After installation is finished, start docker by running this command systemctl start docker.

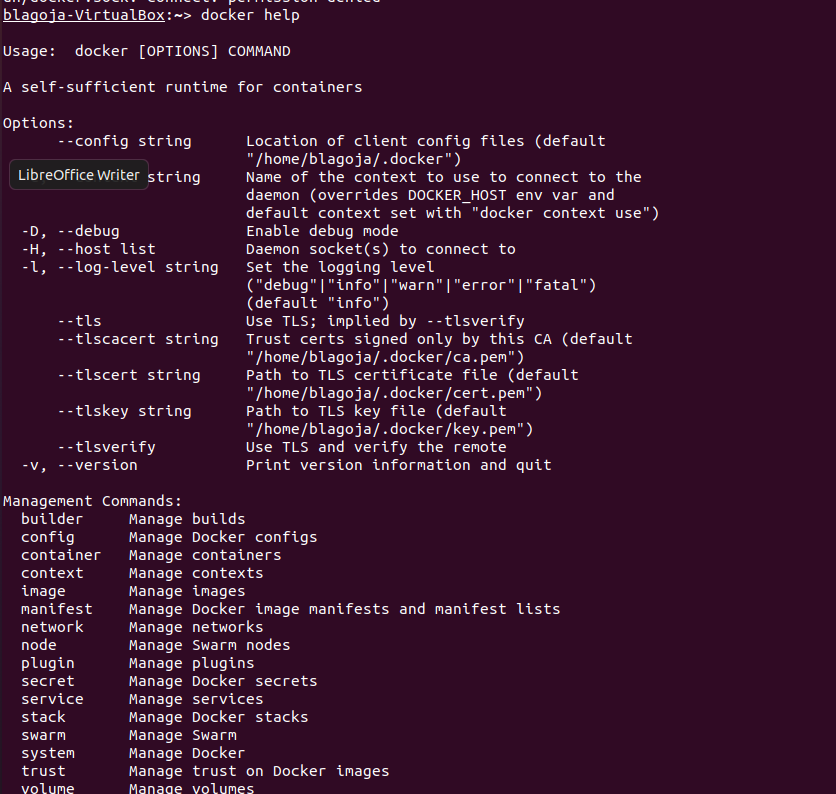
5. Also enable docker service automatic start with command systemctl enable docker.



6. Run docker version to see installed version.

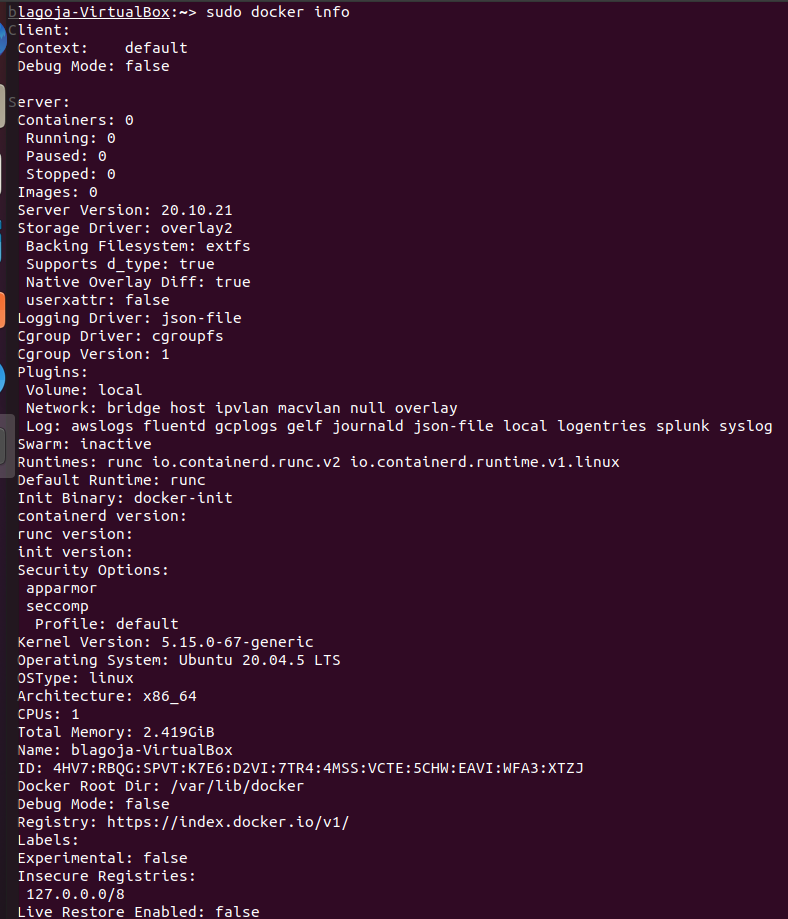


7. Run docker help to see list of available commands.

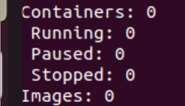


8. Search for a command (switch) that will show system-wide information for your instance of docker.

9. Test it by running docker <command you have discovered>.



10. From the output try to find where the information of number of containers and images is.

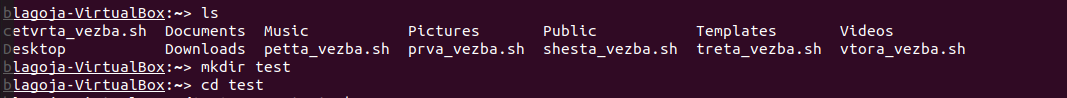


11. Also try to find whether this docker is part of a swarm.



1. Create a Docker container that executes a simple bash script. Go to your home directory and run mkdir test.

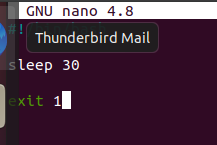
Run cd test



2. Create a simple script. Run vi test.sh.



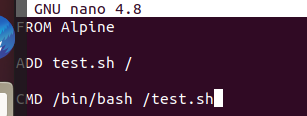
3. Write the following in your script file:



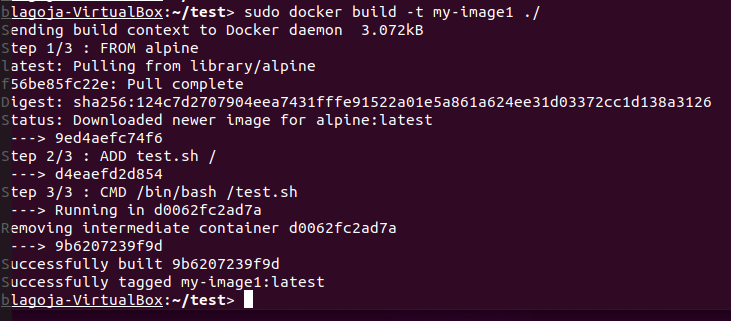
5. Create a docker file. Run vi Dockerfile.



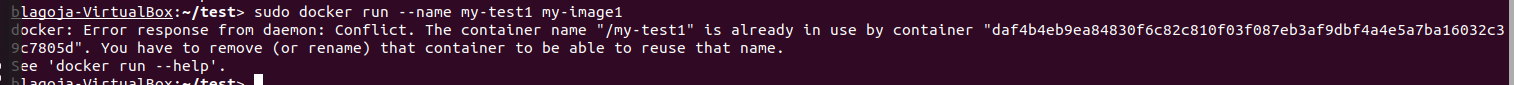
6. Write the following in our Dockerfile:



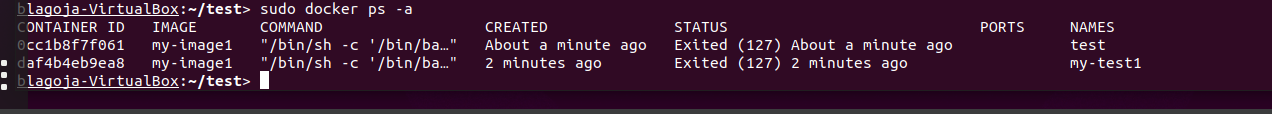
8. Build your image. Run docker build –t my-image1 ./



9. Now spawn a container. Run docker run - -name my-test1 my-image1.



10. Do a docker ps –a. Do you see your container running? \_\_\_\_\_\_\_

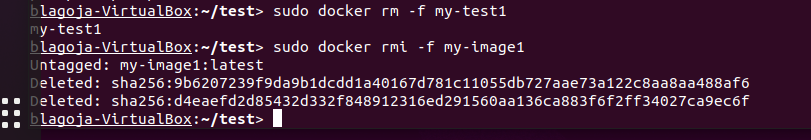


11. Do a docker logs my-test1. What is the output of the log?

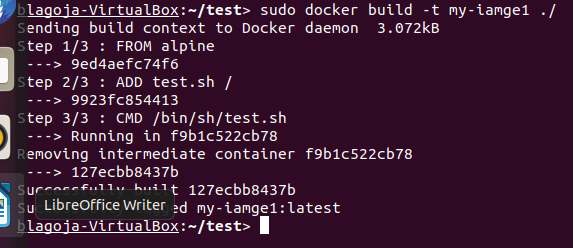


12. Delete my-test. Run docker rm –f my-test1.

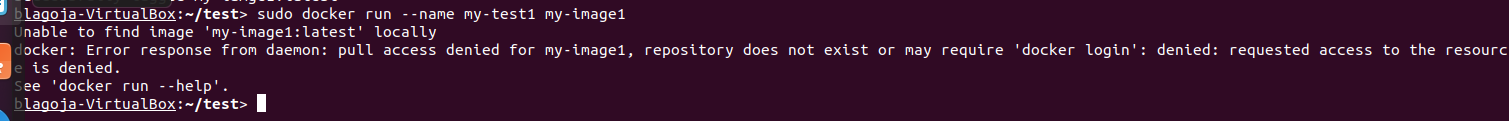
13. Delete my–image. Run docker rmi –f my-image1.



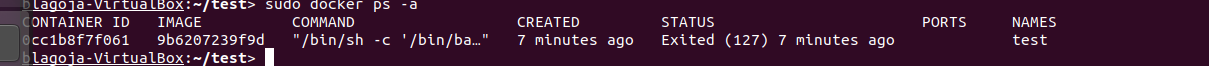
14. Build your image. Run docker build –t my-image1 ./



15. Now spawn a container again. Run docker run - -name my-test1 my-image1.



16. Do a docker ps –a. Do you see your container running?\_\_\_\_\_\_



17. Delete my-test. Run docker rm –f my-test1.

18. Delete my–image. Run docker rmi –f my-image1.

