Cloud Computing

Assignment No. 2

//Paravirtualization: KVM with Virt-Manager

//Install KVM on any Linux platform and alongside install

//virt-manager and run virtualized OS on the KVM through

//virt-manager. Give me step by step process and code

ChatGPT said:

ChatGPT

//Here’s a step-by-step guide to installing KVM (Kernel-based Virtual Machine) on a Linux platform and using Virt-Manager to run a virtualized OS. This guide assumes you are using Ubuntu or a Debian-based distribution.

//Step 1: Check System Compatibility for KVM

//Open the terminal and run:

egrep -c '(vmx|svm)' /proc/cpuinfo

If the output is 0, your CPU does not support hardware virtualization.

If the output is 1 or more, your system supports it.

//Verify that virtualization is enabled in the BIOS.

//Step 2: Install KVM and Required Tools

//Update your system:

sudo apt update && sudo apt upgrade -y

//Install KVM and related packages:

sudo apt install qemu-kvm libvirt-daemon-system libvirt-clients bridge-utils virt-manager -y

//Verify that the libvirtd service is running:

sudo systemctl status libvirtd

//Add your user to the libvirt group to avoid using sudo:

sudo usermod -aG libvirt $(whoami)

//Restart your system or log out and back in for the group changes to take effect.

ubuntu@VaibhavTYB58:~/Desktop$ sudo virt-manager

[sudo] password for ubuntu:

ubuntu@VaibhavTYB58:~/Desktop$ virt-manager

ubuntu@VaibhavTYB58:~/Desktop$

Exp3

//wine

ubuntu@VaibhavTYB58:~/Desktop$ sudo apt update && sudo apt upgrade -y

ubuntu@VaibhavTYB58:~/Desktop$ sudo apt install -y wine64 wine32

ubuntu@VaibhavTYB58:~/Desktop$ wine --version

ubuntu@VaibhavTYB58:~/Desktop$ winecfg

install any windows application(Notepad++) from firefox

ubuntu@VaibhavTYB58:~/Desktop$ wine ~/.wine/drive\_c/Program\ Files/Notepad++/notepad++.exe

DosBOx

ubuntu@VaibhavTYB58:~/Desktop$ sudo apt install -y dosbox

ubuntu@VaibhavTYB58:~/Desktop$ dosbox --version

ubuntu@VaibhavTYB58:~/Desktop$ mkdir ~/dosgames

mario game download link

===https://www.dosgames.com/game/mario-and-luigi/#google\_vignette

Experiment 6

Eavesdropping

https://www.geeksforgeeks.org/working-udp-datagramsockets-java/

//create seperate server and client run it

ubuntu@VaibhavTYB58:~$ sudo apt update

ubuntu@VaibhavTYB58:~$ sudo apt install openjdk-21-jdk

ubuntu@VaibhavTYB58:~$ java -version

ubuntu@VaibhavTYB58:~$ cd Downloads

ubuntu@VaibhavTYB58:~/Downloads$ javac server.java

ubuntu@VaibhavTYB58:~/Downloads$ java server

ubuntu@VaibhavTYB58:~/Downloads$ javac client.java

ubuntu@VaibhavTYB58:~/Downloads$ java client

hello

komal

third terminal:

buntu@VaibhavTYB58:~/Downloads$ sudo tcpdump -i any udp port 1234 -X

[sudo] password for ubuntu:

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on any, link-type LINUX\_SLL (Linux cooked v1), capture size 262144 bytes

17:56:57.315466 IP manager.52578 > manager.1234: UDP, length 5

0x0000: 4500 0021 1796 4000 4011 c2b5 ac10 0430 E..!..@.@......0

0x0010: ac10 0430 cd62 04d2 000d 609f 6865 6c6c ...0.b....`.hell

0x0020: 6f o

17:58:02.069013 IP manager.52578 > manager.1234: UDP, length 5

0x0000: 4500 0021 34a0 4000 4011 a5ab ac10 0430 E..!4.@.@......0

0x0010: ac10 0430 cd62 04d2 000d 609f 6b6f 6d61 ...0.b....`.koma

0x0020: 6c l

Docker-Assignment 8

Microsoft Windows [Version 10.0.22631.4460]

(c) Microsoft Corporation. All rights reserved.

C:\Users\samru>DOCKER --VERSION

unknown flag: --VERSION

See 'docker --help'.

Usage: docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:

run Create and run a new container from an image

exec Execute a command in a running container

ps List containers

build Build an image from a Dockerfile

pull Download an image from a registry

push Upload an image to a registry

images List images

login Authenticate to a registry

logout Log out from a registry

search Search Docker Hub for images

version Show the Docker version information

info Display system-wide information

Management Commands:

builder Manage builds

buildx\* Docker Buildx

compose\* Docker Compose

container Manage containers

context Manage contexts

debug\* Get a shell into any image or container

desktop\* Docker Desktop commands (Alpha)

dev\* Docker Dev Environments

extension\* Manages Docker extensions

feedback\* Provide feedback, right in your terminal!

image Manage images

init\* Creates Docker-related starter files for your project

manifest Manage Docker image manifests and manifest lists

network Manage networks

plugin Manage plugins

sbom\* View the packaged-based Software Bill Of Materials (SBOM) for an image

scout\* Docker Scout

system Manage Docker

trust Manage trust on Docker images

volume Manage volumes

Swarm Commands:

swarm Manage Swarm

Commands:

attach Attach local standard input, output, and error streams to a running container

commit Create a new image from a container's changes

cp Copy files/folders between a container and the local filesystem

create Create a new container

diff Inspect changes to files or directories on a container's filesystem

events Get real time events from the server

export Export a container's filesystem as a tar archive

history Show the history of an image

import Import the contents from a tarball to create a filesystem image

inspect Return low-level information on Docker objects

kill Kill one or more running containers

load Load an image from a tar archive or STDIN

logs Fetch the logs of a container

pause Pause all processes within one or more containers

port List port mappings or a specific mapping for the container

rename Rename a container

restart Restart one or more containers

rm Remove one or more containers

rmi Remove one or more images

save Save one or more images to a tar archive (streamed to STDOUT by default)

start Start one or more stopped containers

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

stop Stop one or more running containers

tag Create a tag TARGET\_IMAGE that refers to SOURCE\_IMAGE

top Display the running processes of a container

unpause Unpause all processes within one or more containers

update Update configuration of one or more containers

wait Block until one or more containers stop, then print their exit codes

Global Options:

--config string Location of client config files (default

"C:\\Users\\samru\\.docker")

-c, --context string Name of the context to use to connect to the

daemon (overrides DOCKER\_HOST env var and

default context set with "docker context use")

-D, --debug Enable debug mode

-H, --host list Daemon socket to connect to

-l, --log-level string Set the logging level ("debug", "info",

"warn", "error", "fatal") (default "info")

--tls Use TLS; implied by --tlsverify

--tlscacert string Trust certs signed only by this CA (default

"C:\\Users\\samru\\.docker\\ca.pem")

--tlscert string Path to TLS certificate file (default

"C:\\Users\\samru\\.docker\\cert.pem")

--tlskey string Path to TLS key file (default

"C:\\Users\\samru\\.docker\\key.pem")

--tlsverify Use TLS and verify the remote

-v, --version Print version information and quit

Run 'docker COMMAND --help' for more information on a command.

For more help on how to use Docker, head to https://docs.docker.com/go/guides/

C:\Users\samru>docker pull helloworld

Using default tag: latest

Error response from daemon: pull access denied for helloworld, repository does not exist or may require 'docker login': denied: requested access to the resource is denied

C:\Users\samru>docker login

Authenticating with existing credentials...

Login Succeeded

C:\Users\samru>docker pull helloworld

Using default tag: latest

Error response from daemon: pull access denied for helloworld, repository does not exist or may require 'docker login': denied: requested access to the resource is denied

C:\Users\samru>docker pull hello-world

Using default tag: latest

latest: Pulling from library/hello-world

c1ec31eb5944: Pull complete

Digest: sha256:305243c734571da2d100c8c8b3c3167a098cab6049c9a5b066b6021a60fcb966

Status: Downloaded newer image for hello-world:latest

docker.io/library/hello-world:latest

C:\Users\samru>docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

C:\Users\samru>docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS

NAMES

9adba4de23b1 dpage/pgadmin4 "/entrypoint.sh" 9 days ago Exited (255) 14 minutes ago 443/tcp, 0.0.0.0:8081->80/tcp pgadmin4-cc-lab

987c934b7b3f node:22-alpine "docker-entrypoint.s…" 9 days ago Exited (1) 9 days ago

b20\_cc\_lab-node-1

9732a74d6685 httpd "httpd-foreground" 9 days ago Exited (0) 9 days ago

httpd-cc-lab

abf6ff9d98ee postgres "docker-entrypoint.s…" 9 days ago Created

postgres-cc-lab

C:\Users\samru>docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

postgres latest 80cbdc6c3301 2 weeks ago 435MB

dpage/pgadmin4 latest da585b3c7d33 4 weeks ago 504MB

node 22-alpine ee9ae20d6258 5 weeks ago 155MB

httpd latest dad6ca1caf78 4 months ago 148MB

hello-world latest d2c94e258dcb 19 months ago 13.3kB

C:\Users\samru>docker run -d -p 8080:80 nginx

Unable to find image 'nginx:latest' locally

latest: Pulling from library/nginx

bc0965b23a04: Pull complete

650ee30bbe5e: Pull complete

8cc1569e58f5: Pull complete

362f35df001b: Pull complete

13e320bf29cd: Pull complete

7b50399908e1: Pull complete

57b64962dd94: Pull complete

Digest: sha256:fb197595ebe76b9c0c14ab68159fd3c08bd067ec62300583543f0ebda353b5be

Status: Downloa^Ad newer image for nginx:latest

b82a318dce0b0bb11064f97fbc08f2a9571d2001349e797c543cfdc7c84b0104

Docker- Assignment 9

**Step 1: Pull the httpd Docker Image**

1. Pull the official Apache HTTP Server (httpd) image:

bash

Copy code

docker pull httpd

1. Verify the image is downloaded:

bash

Copy code

docker images

**Step 2: Run the httpd Container**

1. Start a container:

bash

Copy code

docker run -d --name my-httpd -p 6543:80 httpd

* + -d: Runs the container in detached mode.
  + --name my-httpd: Assigns the container a name.
  + -p 6543:80: Maps **host port 6543** to **container port 80**.

1. Verify the container is running:

bash

Copy code

docker ps

1. Access the webpage:
   * Open your browser and navigate to http://localhost:6543.

**Step 3: Observe Docker Network Configurations**

1. View the Docker networks:

bash

Copy code

docker network ls

You’ll see default networks like bridge, host, and none.

1. Inspect the network used by the httpd container:

bash

Copy code

docker network inspect bridge

This shows details like IP address, subnet, and connected containers.

1. Check the container’s internal network configuration:
   * Use an interactive shell in the container:

bash

Copy code

docker exec -it my-httpd bash

* + Check the network interfaces:

bash

Copy code

ifconfig # Or use `ip addr` if `ifconfig` is unavailable.

* + Exit the container:

bash

Copy code

exit

**Step 4: Create Sample Webpages**

1. Create a folder for your sample webpage:

bash

Copy code

mkdir ~/docker-html

1. Add an HTML file in the folder:

bash

Copy code

echo "<!DOCTYPE html><html><body><h1>My Custom Page</h1></body></html>" > ~/docker-html/index.html

**Step 5: Map the Desktop Path with Docker Volume**

1. Stop the current container:

bash

Copy code

docker stop my-httpd

docker rm my-httpd

1. Start a new container with a volume mapping:

bash

Copy code

docker run -d --name my-httpd -p 6543:80 -v ~/docker-html:/usr/local/apache2/htdocs/ httpd

* + -v ~/docker-html:/usr/local/apache2/htdocs/: Maps the docker-html directory on the host to the web server’s document root inside the container.

1. Access the custom webpage:
   * Open your browser and navigate to http://localhost:6543.
   * You should see **"My Custom Page"**.

**Step 6: Additional Docker Networking and Volume Commands**

* **View container logs:**

bash

Copy code

docker logs my-httpd

* **Inspect the running container:**

bash

Copy code

docker inspect my-httpd

* **Stop and remove containers:**

bash

Copy code

docker stop my-httpd

docker rm my-httpd

* **Clean up unused volumes:**

bash

Copy code

docker volume prune