参考答案

项目一

任务一

一、填空题

1. VMware Workstation
2. CPU　内存　磁盘

二、选择题

1.C　2.D　3.A

三、实训题

略

任务二

一、填空题

1. 大规模虚拟化
2. 2GB　20GB

二、选择题

1. B　2. B　3. D

三、实训题

（1）


活动
终端
8月 6 02：00
zh
xiao-y@xiao-y:~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[xiao-y@xiao-y ~]$ uname -a
Linux xiao-y 4.18.0-526.el8.x86_64 #1 SMP Sat Nov 18 00:54:11 UTC 2023 x86_64 x86_64 x86_64 GNU/Linux
[xiao-y@xiao-y ~]$
Right Ctrl

（2）


手动分区
完成(D)
CENTOS STREAM 8 安装
cn
帮助!
新 CentOS Stream 8 安装
系统
/
sda1
100 GiB
sda1
挂载点(P)：
/
设备：
ATA VBOX HARDDISK (sda)
修改(M)...
期望容量(C)：
100 GiB
设备类型(T)：
标准分区
加密(E)
文件系统(Y)：
xfs
重新格式化(O)
标签(L)：
名称(N)：
sda1
更新设置(U)
注意：在您点击主菜单上的“开始安装”按钮之前，
您在本屏幕内所做的设置更改不会被应用。
可用空间
20 GiB
总空间
120 GiB
已选择 1 个存储设备(S)
全部重设(R)


活动
磁盘
8月5 23：54
zh
129 GB 硬盘
/dev/sda
129 GB 硬盘
VBOX HARDDISK
CD/DVD 驱动器
CD-ROM
型号 VBOX HARDDISK (1.0)
大小 129 GB (128,849,018,880 字节)
分区 主引导记录
序列号 VBfd86cfd6-c5bedc38
评估 不支持 SMART
卷(V)
文件系统
分区 1
107 GB XFS
剩余空间
21 GB
大小 107 GB — 101 GB 可用（已用 6.0%）
设备 /dev/sda1
UUID f98c08a6-ec01-4415-8e7c-022e264ca8f4
分区类型 Linux (可启动)
内容 XFS — 已挂载到 文件系统根目录

（3）


活动
设置
8月5 23：54
zh
详细信息
关于
关于
日期和时间
用户
默认应用程序
CentOS Stream 8
设备名称 xiao-y
内存 1.7 GiB
处理器 12th Gen Intel® Core™ i5-12500
图形 llvmpipe (LLVM 16.0.6, 256 bits)
GNOME 版本 3.32.2
操作系统类型 64 位
虚拟化 Oracle
磁盘 107.3 GB
(V)查看更新

（4）

项目二

任务一

一、填空题

1. 复制文件 test.txt 为 mytest.txt 文件
2. :q
3. wc -l a.txt >> output
4. mv file1.c file2.c
5. rm -rf /home/tmp

二、选择题

1. A　2. B　3. A　4. C　5. C

三、实训题

（1）

```
mkdir /path/to/mydir
touch /path/to/mydir/myfile.txt
```

（2）

```
cd /path/to/mydir
mkdir subdir
cd subdir
touch file1.txt file2.txt
```

（3）

```
cd /path/to/mydir
cd subdir
```

（4）

```
ls -a /path/to/mydir
```

（5）

```
cp subdir/file1.txt mydir/newfile.txt
```

任务二

一、填空题

1. 普通用户　超级用户（root）
2. 读　/etc/shadow
3. /etc/passwd
4. chmod
5. 读写

二、选择题

1. C　2. D　3. C　4. D　5. C

三、实训题

（1）

```
groupadd deve
groupadd Oper
useradd -g deve d001
useradd -g deve d002
useradd -g Oper p001
useradd -g Oper p002
```

（2）

```
mkdir /public
chmod 777 /public
chgrp Oper /public
```

（3）

```
su d001
cd /public
touch file_devp
exit

su p001
cd /public
touch file_oper
exit
```

（4）

```
su d001
cd /public
rm file_oper
ls -l
```

（5）

```
groupdel deve
groupdel Oper
userdel d001
userdel d002
userdel p001
userdel p002
rm -r /public
```

任务三

一、填空题

1. 写
2. chmod +x filename
3. 读写
4. 符号
5. su

二、选择题

1. D　2. B　3. D　4. C　5. C

三、实训题

（1）

```
groupadd deve
groupadd Oper
groupadd net
useradd -g deve d001
useradd -g net p001
```

（2）

```
方法一：修改文件权限
chmod 777 /home/d001/test_file
方法二：修改文件属组
chown p001 /home/d001/test_file
方法三：将 p001 用户加入 d001 组
usermod -a -G d001 p001
```

项目三

任务一

一、填空题

1. df -h
2. fdisk
3. /etc/fstab
4. mkfs.ext4
5. lsblk

二、选择题

1. B　2. A　3. C　4. C　5. B

三、实训题

（1）

```
lsblk
```

（2）

```
fdisk /dev/sdX
```

（3）

```
mkfs.ext4 /dev/sdX1
```

（4）

```
mkdir /mnt/data
mount /dev/sdX1 /mnt/data
```

（5）

```
echo "/dev/sdX1 /mnt/data ext4 defaults 0 2" >> /etc/fstab
```

任务二

一、填空题

1. 逻辑卷管理（Logical Volume Manager）
2. 磁盘 分区
3. lvextend
4. lvreduce
5. 卷组（VG）

二、选择题

1. A　2. A　3. B　4. C　5. C

三、实训题

（1）

```
pvcreate /dev/sdb /dev/sdc
```

（2）

```
vgcreate app_vg /dev/sdb /dev/sdc
```

（3）

```
lvcreate -L 2T -n app_lv app_vg
```

（4）

```
mkfs.xfs /dev/app_vg/app_lv
```

（5）

```
mkdir /app
mount /dev/app_vg/app_lv /app
df -h
lsblk
```

项目四

任务一

一、填空题

1. 静态 IP 地址
2. /etc/dhcp/dhcpd.conf
3. subnet
4. systemctl start dhcpd
5. systemctl enable dhcpd

二、选择题

1. A　2. A　3. B

三、实训题

略

任务二

一、填空题

1. /var/named/chroot
2. UDP
3. hosts
4. 将无法解析的域名请求转发给其他 DNS 服务器进行处理（提高 DNS 解析的效率和准确性，减轻 DNS 服务器的负载）
5. nsupdate

二、选择题

1. A　2. A　3. A

三、实训题

略

任务三

一、填空题

1. 路由 防火墙
2. root sudo yum install -y iptables
3. sysctl -p

二、选择题

1. A　2. A　3. A

三、实训题

（1）打开终端，使用如下命令。

```
sudo yum install -y iptables iproute2
```

（2）需要编辑/etc/sysconfig/network-scripts/目录下的网络接口配置文件，通常文件名以 ifcfg-开头，后跟接口名称。

```
sudo vim/etc/sysconfig/network-scripts/ifcfg-ens160
```

然后确保以下参数被正确设置。

```
BOOTPROTO=static
IPADDR=你的静态 IP 地址
NETMASK=子网掩码
GATEWAY=网关地址
```

（3）编辑/etc/sysctl.conf 文件。

```
sudo vim /etc/sysctl.conf
  net.ipv4.ip_forward = 1
  sudo sysctl -p
```

（4）配置 NAT 规则，允许内网设备通过软路由访问外网。

```
sudo iptables -t nat -A POSTROUTING -o ens160 -j MASQUERADE
  sudo iptables -A FORWARD -i ens160 -o ens33 -j ACCEPT
  sudo iptables -A FORWARD -i ens33 -o ens160 -j ACCEPT
  sudo iptables-save > /etc/iptables/rules.v4
```

（5）安装 dhcp-server，编辑配置文件/etc/dhcp/dhcpd.conf，设置 IP 地址池、子网掩码、网

关等信息。

```
    sudo yum install dhcp-server -y
    sudo vim /etc/dhcp/dhcpd.conf
# 配置示例
     subnet 192.168.0.0 netmask 255.255.255.0 {
          range 192.168.0.100 192.168.0.200;
          option routers 192.168.0.1;
          option domain-name-servers 8.8.8.8;
     }
     sudo systemctl start dhcpd
     sudo systemctl enable dhcpd
```

（6）编辑 iptables 规则，允许 DHCP 流量。

```
sudo iptables -A INPUT -p udp --dport 67:68 -j ACCEPT
```

（7）启动 dhcpd 服务，确保客户端设备设置为自动获取 IP 地址。

```
sudo systemctl start dhcpd
sudo systemctl enable dhcpd
```

（8）检查 dhcpd 的日志。

```
    sudo tail -f /var/log/messages
```

以上步骤完成后，需要提交包含配置文件、命令执行的截图，截图略。

项目五

任务一

一、填空题

1. 防火墙 网络安全
2. 硬件防火墙　软件防火墙　基于云的防火墙
3. systemctl stop firewalld
   systemctl disable firewalld

二、选择题

1. C　2. D　3. B　4. C　5. D　6. C

三、实训题

略

任务二

一、填空题

1. 区域（zone） 区域 区域
2. /etc/firewalld/zones/
   /etc/firewalld/firewalld.conf

二、选择题

1. C 2. C 3. A

三、实训题

```
iptables -I INPUT -s 10.0.0.188 -p tcp --dport 80 -j REJECT
```

项目六

任务一

一、填空题

1.客户-服务器
2. /etc/httpd/conf/httpd.conf
3. UserDir
4. DocumentRoot
5. /etc/httpd/conf

二、选择题

1. A 2. D 3. C 4. B 5. B

三、实训题

（1）

```
 sudo dnf update -y
sudo dnf install -y httpd
```

（2）

```
sudo vi /var/www/html/index.html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Hello World</title>
</head>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

（3）在 CentOS 8 上，Apache 的默认网站目录是 /var/www/html。在上一步中已经将 index.html 文件放置在这个目录中了。

（4）

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

（5）确保防火墙允许 HTTP 流量（端口 80）。然后，在浏览器中输入服务器的 IP 地址或域名，就能看到 "Hello, World!" 的页面。

```
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --reload
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

任务二

一、填空题

1. 实际物理
2. all granted
3. Apache
4. 客户/服务器
5. 真实物理目录

二、选择题

1. B　2. A　3. A　4. A　5. B

三、实训题

```
Alias /images /var/images
<Directory /var/images>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

任务三

一、填空题

1. service vsftpd start
2. 主动模式 被动模式
3. mget
4. 21
5. anonymous

二、选择题

1. D　2. A　3. C　4. D　5. C

三、实训题

（1）

```
yum install vsftpd    -y
```

（2）

```
adduser ftpuser
```

（3）

```
local_enable=YES
write_enable=YES
chroot_local_user=YES
local_root=/var/ftp
```

（4）

```
service vsftpd start
```

任务四

一、填空题

1. anonymous
2. anonymous
3. service vsftpd start
4. mput
5. 20 21

二、选择题

1. D　2. A　3. C　4. C　5. C

三、实训题

（1）

```
# 在 vsftpd 配置文件 /etc/vsftpd.conf 中添加或修改以下配置
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES

# 重启 vsftpd 服务使配置生效
service vsftpd restart
```

（2）

```
# 在 vsftpd 配置文件 /etc/vsftpd.conf 中添加或修改以下配置
user_restrict_enable=YES
user_restrict_file=/etc/vsftpd/user_list

# 在 /etc/vsftpd/user_list 文件中添加用户及其权限配置
user1
user2 write

# 重启 vsftpd 服务使配置生效
service vsftpd restart
```

（3）

```
# 在 vsftpd 配置文件 /etc/vsftpd.conf 中添加或修改以下配置
chroot_local_user=YES
local_root=/home
```

```
# 重启 vsftpd 服务使配置生效
service vsftpd restart
```

项目七

任务一

一、填空题

1. 容器
2. 容器和容器镜像
3. 开
4. 基于 OCI 的规范和标准
5. 守护进程

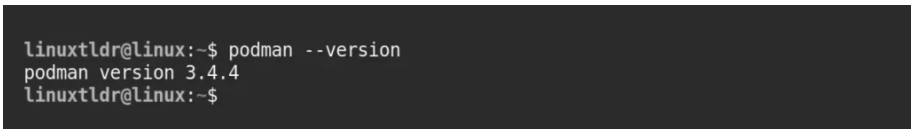
二、选择题

1. C　2. B　3. D　4. A

三、实训题

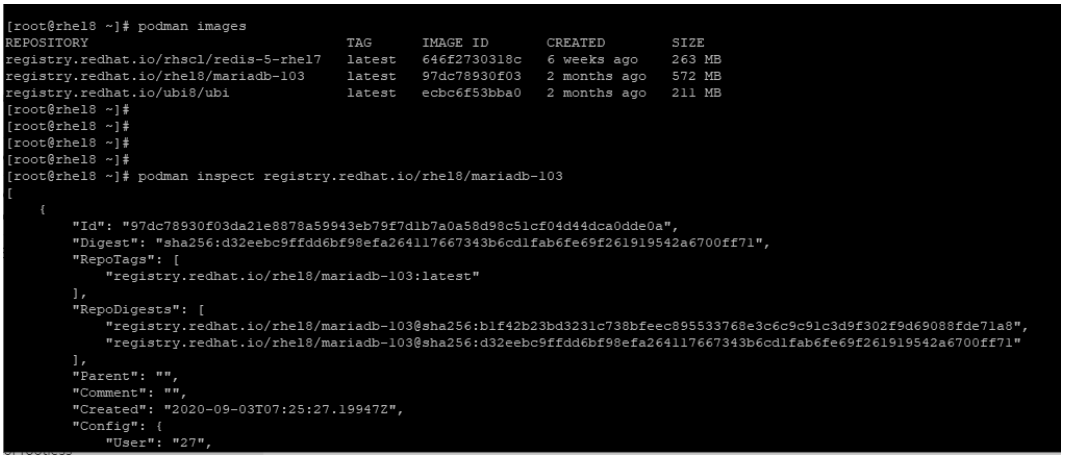
（2）安装 Podman

```
sudo dnf install -y podman
podman　–version
```



（3）配置 Podman

```
podman pull [OPTIONS] NAME[:TAG|@DIGEST]
podman pull ubuntu:latest
podman images
```



```
[root@rhel8 ~]# podman images
REPOSITORY                                TAG      IMAGE ID       CREATED        SIZE
registry.redhat.io/rhscl/redis-5-rhel7    latest   646f2730318c   6 weeks ago    263 MB
registry.redhat.io/rhel8/mariadb-103      latest   97dc78930f03   2 months ago   572 MB
registry.redhat.io/ubi8/ubi               latest   ecbc6f53bba0   2 months ago   211 MB
[root@rhel8 ~]#
[root@rhel8 ~]#
[root@rhel8 ~]#
[root@rhel8 ~]#
[root@rhel8 ~]# podman inspect registry.redhat.io/rhel8/mariadb-103
[
    {
        "Id": "97dc78930f03da21e8878a59943eb79f7d1b7a0a58d98c51cf04d44dca0dde0a",
        "Digest": "sha256:d32eebc9ffdd6bf98efa264117667343b6cd1fab6fe69f261919542a6700ff71",
        "RepoTags": [
            "registry.redhat.io/rhel8/mariadb-103:latest"
        ],
        "RepoDigests": [
            "registry.redhat.io/rhel8/mariadb-103@sha256:b1f42b23bd3231c738bfeec895533768e3c6c9c91c3d9f302f9d69088fde71a8",
            "registry.redhat.io/rhel8/mariadb-103@sha256:d32eebc9ffdd6bf98efa264117667343b6cd1fab6fe69f261919542a6700ff71"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2020-09-03T07:25:27.199947Z",
        "Config": {
            "User": "27",
```

export CONTAINER_REGISTRY_MIRROR=https://cr.console. aliyun.com/

sudo nano /etc/containers/containers.conf

[registries.insecure] registries = ["https://your-mirror"]

podman run -it --rm --network mynamespace ubuntu /bin/bash

podman run -it --rm --network host ubuntu /bin/bash

创建桥接

sudo ip link add br0 type bridge

sudo ip addr add 192.168.0.1/24 dev br0

sudo ip link set dev br0 up

运行容器并连接到桥接

podman run -it --rm --network bridge:br0 ubuntu /bin/bash

在容器中配置网络

ip link set dev eth0 up ip addr add 192.168.0.2/24 dev eth0

编写 YAML 文件来定义多个容器之间的网络配置。

下载 Podman Compose

curl -L https://github.com/containers/podman-compose/releases/latest

/download/podman-compose -o /usr/local/bin/podman-compose

授予执行权限

chmod +x /usr/local/bin/podman-compose

使用 nano 编辑器打开“docker-compose.yaml”文件

nano docker-compose.yaml

编辑 YAML 文件。

version: '3'

services:

  web:

    image: nginx

    ports:

```
            - "8080:80"
        db:
            image: postgres
            environment:
                POSTGRES_PASSWORD: example
    # 在 Podman Compose 文件所在的目录中运行以下命令：
    podman-compose up
    # 停止和清理
    # 要停止运行的服务，可以运行：
    podman-compose down
    # 在运行 Rootless 模式之前，请确保系统支持该模式。您可以通过以下命令检查：
    podman –version
    # Rootless 模式通常需要启用用户命名空间，以确保容器在非特权用户下运行。在大多数系统上，用户命名空间默认是启用的，但我们可以检查一下。
    cat /proc/sys/kernel/unprivileged_userns_clone
    # 如果输出为 0，表示用户命名空间未启用。您可以通过以下方式启用：
    echo "kernel.unprivileged_userns_clone=1"
    sudo tee -a /etc/sysctl.conf
    sudo sysctl --system
    # 配置 Rootless Podman
    export PATH=$PATH:$HOME/.local/bin
    export CONTAINER_SOCK=/run/user/$(id -u)/podman/podman.sock
    source ~/.bashrc
    # 启动 Rootless Podman
    podman system service --time=0 tcp://localhost:12345
    # 测试 Rootless 模式
    podman run --rm -it ubuntu /bin/bash
```

任务二

一、填空题

1. podman run
2. --name
3. podman exec
4. podman ps
5. podman rm

二、选择题

1. B　2. C　3. A　4. B　5. D

三、实训题

```
# 查找容器镜像
podman search ubuntu
```

```
temporary@ubuntu:~/workdir$ podman search ubuntu --filter=is-official
INDEX       NAME                                    DESCRIPTION                                     STARS   OFFICIAL   AUTOMATED
docker.io   docker.io/library/ubuntu                Ubuntu is a Debian-based Linux operating sys...   12306   [OK]
docker.io   docker.io/library/ubuntu-debootstrap    debootstrap --variant=minbase --components=m...   44      [OK]
docker.io   docker.io/library/ubuntu-upstart        Upstart is an event-based replacement for th...   110     [OK]
docker.io   docker.io/library/websphere-liberty     WebSphere Liberty multi-architecture images ...   273     [OK]
docker.io   docker.io/library/open-liberty          Open Liberty multi-architecture images based...   45      [OK]
temporary@ubuntu:~/workdir$
```

```
podman pull docker.io/library/ubuntu
podman run -it --name my_ubuntu docker.io/library/ubuntu /bin/bash
# 查找 Python 官方镜像
podman search python
podman pull docker.io/library/python
podman run -it --name web_python -p 8080:80 docker.io/library/python:latest
python -m http.server 80
```

启动容器

podman start my_container

查看正在运行的容器，要查看当前正在运行的容器

podman ps

停止容器

podman stop my_ubuntu

重新启动容器

podman stop my_ubuntu

删除容器

podman rm my_ubuntu

性能调优

podman run -it --rm --memory 512m ubuntu /bin/bash

创建自定义桥接网络

步骤一：创建桥接网络。

sudo ip link add br_custom type bridge

sudo ip addr add 192.168.0.1/24 dev br_custom

sudo ip link set dev br_custom up

步骤二：运行容器并连接到桥接网络。

podman run -it --rm --network bridge:br_custom ubuntu /bin/bash

步骤三：在容器中配置网络：

ip link set dev eth0 up ip addr add 192.168.0.2/24 dev eth0

管理容器端口映射

步骤一：运行带有端口映射的容器

podman run -d --name web_server -p 8080:80 nginx

步骤二：访问映射的端口

连接到主机网络

podman run -it --rm --network host ubuntu /bin/bash

任务三

一、填空题

1. 在 Podman 上定义和管理多容器应用程序
2. 应用容器
3. YAML
4. 通信

二、选择题

1. A　2. A　3. B　4. A

三、实训题

（1）Podman Compose 文件结构

```
version: '3'      # Compose 文件版本

services:    # 定义服务
web:      # 服务名称
image: nginx:latest      # 使用的镜像
ports:
- "8080:80"      # 端口映射
networks:      # 定义网络
app_net:
driver: bridge      # 使用的网络驱动
ipam:           # IP 地址管理
driver: default
config:
- subnet: "172.18.0.0/24"
```

（2）定义服务

```
services:
webapp:
image: my_web_app:latest
ports:
- "8080:80"
environment:
- NODE_ENV=production
- DB_HOST=database
volumes:
- ./app:/app
depends_on:
- database
```

（3）定义网络

```
networks:
backend_network:
driver: bridge
ipam:
driver: default
config:
- subnet: "172.19.0.0/24"
```

（4）定义卷

```
volumes:
data_volume:
```

使用 Podman Compose 启动应用

podman-compose up
（5）扩展性
services:
base:
image: ubuntu:latest
environment:
- COMMON_VARIABLE=common_value
app:
extends:
service: base
environment:
- APP_VARIABLE=app_value
多文件配置
docker-compose -f docker-compose.base.yml -f docker-compose.prod.yml up
Compose 使用别名
services: web: image: nginx:latest container_name: my_nginx
Compose 覆盖默认配置
podman-compose -f podman-compose.yml -f podman-compose.override.yml up

Docker Compose 管理多容器文件基础
version: '3' # Compose 文件版本

services: # 定义服务
web: # 服务名称
image: nginx:latest # 使用的镜像
ports:
- "8080:80" # 端口映射

database: # 另一个服务
image: postgres:latest
environment:
POSTGRES_PASSWORD: example_password
多容器服务之间的连接
version: '3'

services:
web:
image: my_web_app:latest
ports:

```
- "8080:80"
depends_on:
- database
database:
image: postgres:latest
environment:
POSTGRES_PASSWORD: example_password
多容器网络
version: '3'

services:
web:
image: my_web_app:latest
ports:
- "8080:80"
networks:
- my_network
database:
image: postgres:latest
environment:
POSTGRES_PASSWORD: example_password
networks:
- my_network

networks:
my_network:
driver: bridge
多容器卷的管理
version: '3'

services:
web:
image: my_web_app:latest
ports:
- "8080:80"
volumes:
- my_data:/app/data

volumes:
```

my_data:

多容器环境变量的使用

version: '3' services: web: image: my_web_app:latest ports: - "8080:80" environment: NODE_ENV: production DATABASE_URL: postgres://user:password@database:5432/db API_KEY: abc123 database: image: postgres:latest environment: POSTGRES_PASSWORD: example_password

多容器多文件配置

podman-compose -f podman -compose.base.yml -f podman -compose.prod.yml up

使用 Podman Compose 启动应用、

podman -compose up

项目八

任务一

一、填空题

1. 无守护进程的容器引擎
2. 一个完整的、基于云的虚拟桌面环境
3. 容器
4. 降低成本　提高安全性　提升灵活性和效率

二、选择题

1. B　2. C　3. B　4. D

三、实训题

podman pull ubuntu:20.04

podman run -d --name cloud-desktop -p 5901:5901 -e DISPLAY=:1 ubuntu:20.04

podman exec -it cloud-desktop /bin/bash

安装桌面环境

apt update

apt install xfce4 xfce4-goodies

配置 Xfce 桌面环境

xfconf-query -c xfce4-desktop -p /backdrop/screen0/monitor0/workspace0/last-image -s /path/to/your/image.jpg

xfconf-query -c xfce4-desktop -p /backdrop/screen0/monitor0/workspace0/

color1-s "color_value"

使用命令行工具调整面板位置和大小：

xfconf-query -c xfce4-panel -p /panels/panel-1/position -s "p=10;x=100;y=100" xfconf-query -c xfce4-panel -p /panels/panel-1/length -s "size_value"

添加应用程序启动器

xfconf-query -c xfce4-panel -p /panels/panel-1/objects -t string -s "launcher-1" -a

xfconf-query -c xfce4-panel -p /panels/panel-1/objects/launcher-1 -t string -s "xfce4-terminal.desktop"

使用脚本自定义启动器图标：

```
#!/bin/bash
# Set custom icon for launcher
xfconf-query -c xfce4-panel -p /plugins/plugin-1/icon -s /path/to/your/icon.png
```

使用命令行工具设置窗口管理器样式：

```
xfconf-query -c xfwm4 -p /general/theme -s "your_theme_name"
```

保存容器状态

```
podman commit cloud-desktop my-cloud-desktop-image
```

编写 Podmanfile

```
RUN apt-get update && apt-get install -y \
    xfce4 \
    xfce4-goodies \
    tightvncserver \
    firefox \
    && apt-get clean

# 设置 VNC 密码
RUN mkdir -p ~/.vnc && echo "password" | vncpasswd -f > ~/.vnc/passwd

# 启动 VNC 服务器
CMD ["vncserver", "-fg"]
```

构建镜像

```
Podman build -t ubuntu-desktop
```

启动容器

```
#!/bin/bash

Podman run -d -p 5901:5901 --name ubuntu-desktop-container ubuntu-desktop
```

运行启动脚本

```
chmod +x start_container.sh
```

```
./start_container.sh

sudo apt update sudo apt install firefox

Wget  https://dl.google.com/linux/direct/google-chrome-stable_current_amd
64.deb
sudo dpkg -i google-chrome-stable_current_amd64.deb

安装 Visual Studio Code
# 从官方网站下载安装包
wget https://go.microsoft.com/fwlink/?LinkID=760868 -O vscode.deb

# 安装 Visual Studio Code
sudo dpkg -i vscode.deb

# 安装依赖项
sudo apt-get install -f
```

任务二

一、填空题

1. 创建用户账户　修改用户信息　重置用户密码　删除用户账户
2. 密码认证　网络级别身份验证（NLA）　双因素身份验证（2FA）
3. 访问权限　操作权限　数据权限　管理权限
4. 登录活动　操作行为　会话活动
5. 核心

二、选择题

1. D　2. A　3. D　4. D　5. A

三、实训题

步骤一：运行容器并映射端口
在运行容器时，映射 VNC 服务器的端口。
podman run -p 5901:5901 -it my-desktop-app:latest
步骤二：更新软件包索引
更新软件包索引
sudo apt update

步骤三：使用以下命令安装 TightVNC Server：

sudo apt install tightvncserver

步骤四：启动 VNC 服务器

步骤五：配置 VNC 服务器

nano ~/.vnc/xstartup

配置文件内容

#!/bin/bash

xrdb $HOME/.Xresources

startxfce4 &

步骤六：启动 VNC 服务器

tightvncserver -kill :1

tightvncserver :1

配置防火墙：

sudo ufw allow 5901/tcp

用户管理

添加用户，在 Ubuntu 中，可以使用 adduser 命令来添加新用户：

sudo adduser username

删除用户，删除用户可以使用 deluser 命令：
sudo deluser username
修改用户密码：
sudo passwd username
用户组管理：
sudo addgroup groupname
将用户添加到组：
sudo adduser username groupname
从组中删除用户：
sudo deluser username groupname
文件权限管理：
chmod permissions filename
文件所有者和组，可以使用 chown 命令更改文件的所有者和组：
chown owner:group filename
sudo 权限管理：
sudo usermod -aG sudo username
修改 sudo 权限：
sudo visudo
SSH 登录管理：
PasswordAuthentication no