**MODUL EXAM 200**

**Note:**

On Servera: Exam Task 1,2,3,4,5,6,7,8,9,11,12,13,14,15,16,18

On Serverb: Exam Task 10,17,19

Commad grade: nusactl grade rhadm-exam-(no.exam)

Disclaimer: The answers to this practice exam are made privately based on direct practice and research results from several sources there is a possibility that the answers can be different or less precise.

**Task 00 - Gaining root access**

1. Open remote viewer
2. Reboot (Ctrl+Alt+Del)
3. Click e
4. Type 'rd.break' at the end of the line
5. Ctrl-x
6. mount -o remount rw /sysroot
7. chroot /sysroot
8. passwd
9. touch /.autorelabel
10. exit
11. exit

**Task 01 - Manage local user and group**

|  |
| --- |
| 1. On servera, add 3 users: **harry, natasha, tom**. 2. The requirements: The Additional group of the two users: **harry**, **natasha** is the **admin** group. 3. The user: **tom's** login shell should be non-interactive. |

1. useradd -G admin harry
2. useradd -G admin natasha
3. useradd -s /sbin/nologin tom
4. cat /etc/group

**DESCRIPTION**

**user/groupadd:** This command is used to add a new user account/group

**user/groupdel:** This command is used to delete a user account/group

**passwd:** This command is used to set or change the password for a user.

**gpasswd:** This command is used to manage group passwords and group membership.

**User/groupmod:** This command is used to modify user account/group properties.

**chage:** This command is used to set password aging policies for a user.

**newgrp:** This command is used to start a new shell session with a different group identity.

**id:** This command displays user and group information.

**variables & example:**

**-c, --comment** : sudo usermod -c "John Doe" username

**-d, --home** :sudo usermod -d /new/home/directory username

**-e, --expiredate :** sudo usermod -e YYYY-MM-DD username

**-g, --gid** : sudo usermod -g new\_primary\_group username

**-G, --groups :** sudo usermod -G group1,group2 username

**-l, --login** : sudo usermod -l newusername oldusername

**-s, --shell** : sudo usermod -s /bin/bash username

**-u, --uid** : sudo usermod -u new\_uid username

**-L, --lock**  : sudo usermod -L username

**-U, --unlock** :sudo usermod -U username

**-p, --password** : sudo usermod -p encrypted\_password username

**Exam Task 02 - Tuning Systems**

Change the current tuning profile for **servera** to **virtual-guest**, a general non-specialized tuned profile.

1. tuned-adm list
2. tuned-adm profile virtual-guest
3. tuned-adm active

**Table Tuning Profiles Distributed with Red Hat Enterprise Linux 8**

| **Tuned Profile** | **Purpose** |
| --- | --- |
| balanced | Ideal for systems that require a compromise between power saving and performance. |
| desktop | Derived from the balanced profile. Provides faster response of interactive applications. |
| throughput-performance | Tunes the system for maximum throughput. |
| latency-performance | Ideal for server systems that require low latency at the expense of power consumption. |
| network-latency | Derived from the latency-performance profile. It enables additional network tuning parameters to provide low network latency. |
| network-throughput | Derived from the throughput-performance profile. Additional network tuning parameters are applied for maximum network throughput. |
| powersave | Tunes the system for maximum power saving. |
| oracle | Optimized for Oracle database loads based on the throughput-performance profile. |
| virtual-guest | Tunes the system for maximum performance if it runs on a virtual machine. |
| virtual-host | Tunes the system for maximum performance if it acts as a host for virtual machines. |

**Task 03 - Enabling Software Repositories**

|  |
| --- |
| On **servera**, set local repositories from url :   * **http://content.example.com/rhel8.0/x86\_64/dvd/BaseOS** * **http://content.example.com/rhel8.0/x86\_64/dvd/AppStream**. |

1. vi /etc/yum.repos.d/local.repo

[localrepo-baseos]

name=Local Repository - BaseOS

baseurl=http://content.example.com/rhel8.0/x86\_64/dvd/BaseOS

enabled=1

gpgcheck=0

[localrepo-appstream]

name=Local Repository - AppStream

baseurl=http://content.example.com/rhel8.0/x86\_64/dvd/AppStream

enabled=1

gpgcheck=0

1. yum clean all
2. yum repolist

**DESCRIPTION**

* Enable a repository: sudo yum-config-manager --enable repository\_name
* Disable a repository: sudo yum-config-manager --disable repository\_name
* Add a repository: sudo yum-config-manager --add-repo repository\_url
* Remove a repository: sudo yum-config-manager --remove repository\_name

**/etc/yum.repos.d/** This directory contains repository configuration files. You can manually edit these files to add or modify repositories.

**Task 04 - Access Network-attached Storage (Autofs)**

**serverb** shares the **/shares** directory using NFS. Mount it to mountpoint **/mnt/shares** on **servera**

1. mkdir /mnt/shares
2. mount -t nfs serverb:/shares /mnt/shares

**DESCRIPTION**

1. **Mounting NFS Shares:**

**mount:** You can use the mount command to mount NFS shares.

sudo mount -t nfs server\_ip:/path/to/share /mnt/mount\_point

**/etc/fstab:** For permanent mounts, add an entry in the /etc/fstab file.

server\_ip:/path/to/share /mnt/mount\_point nfs defaults 0 0

**umount:** To unmount an NFS share.

sudo umount /mnt/mount\_point

1. **Autofs:** Autofs can be used to automatically mount NFS or SMB/CIFS shares when accessed.

**/etc/auto.master:** Configure Autofs in the /etc/auto.master file.

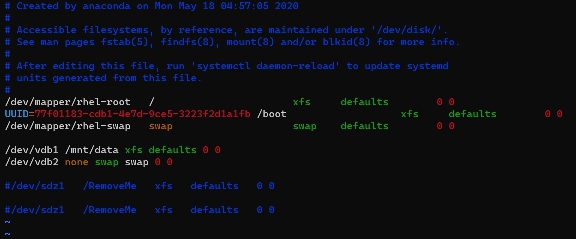
/mnt/mount\_point /etc/auto.nfs

**/etc/auto.nfs:** Create a configuration file for Autofs.

share\_name -fstype=nfs server\_ip:/path/to/share

**Task 05 - Managing Basic Storage**

On **servera**, create gpt partition 200MB (xfs) from disk **/dev/vdb** and mount to **/mnt/data**. Make it persistent !

1. lsblk
2. parted /dev/vdb mklabel gpt
3. parted >print /dev/vdb **(check)**
4. parted /dev/vdb mkpart primary xfs 0% 200MB
5. mkfs.xfs /dev/vdb1
6. mkdir /mnt/data
7. mount /dev/vdb1 /mnt/data
8. vi /etc/fstab
9. /dev/vdb1 /mnt/data xfs defaults 0 0 (dibawah)

**DESCRIPTION**

**Start parted: Open parted for a specific disk.**

**sudo parted /dev/sdX**

**Print: Display the current partition table of the selected disk.**

**Create a New Partition: Interactively create a new partition.**

**mkpart** **[primary|extended|logical] [fs-type] start end**

**Delete Partition: Delete a partition.**

**rm partition\_number**

**Resize Partition: Resize a partition.**

**resizepart partition\_number end**

**Set Partition Label: Assign a label to a partition.**

**name partition\_number label**

**Set Partition Filesystem: Set or change the filesystem of a partition.**

**mkfs partition\_number filesystem-type**

**Toggle Partition Flag: Change partition attributes (e.g., boot flag).**

**set partition\_number [flag]**

**Move Partition: Move a partition to a different location on the disk.**

**move partition\_number new\_start**

**EFFICIENT PARTED COMMANDS**

**Check Disk:** sudo parted /dev/sdX print | grep "Disk /dev/sdX:"

**Create Partition:** sudo parted /dev/sdX mkpart [primary|extended|logical] [fs-type] start end

**Resize Partition:** sudo parted /dev/sdX resizepart partition\_number size

**Partition Label**: sudo parted /dev/sdX name partition\_number DataPartition

**Format Partition (ext4):** sudo mkfs.ext4 /dev/sdXY

**Change Partition** **Filesystem:** sudo parted /dev/sdX mkfs partition\_number [Filesystem]

**Delete Partition:** sudo parted /dev/sdX rm partition\_number

**Task 06 - Managing Basic Storage**

On **servera**, create gpt partition 500MB from disk **/dev/vdb** and set to **swap**.

1. parted /dev/vdb mkpart primary linux-swap 201MB 700MB
2. mkswap /dev/vdb2
3. swapon /dev/vdb2
4. vim /etc/fstab
5. /dev/vdb2 none swap swap 0 0

**DESCRIPTION**

**ALL COMMANDS TO CREATING AND MANAGING A SWAP PARTITION:**

**Create:** sudo parted /dev/sdX mkpart primary linux-swap start end

**Format:** sudo mkswap /dev/sdXY

**Active:** sudo swapon /dev/sdXY

**Deactive:** sudo swapoff /dev/sdXY

**Print Active:** swapon --show

**Add to /etc/fstab:** /dev/sdXY none swap defaults 0 0

**Task 07 - Implementing Advanced Storage Feature**

On **servera**, create 500MB logical volume with name **lv0** using **xfs** filesystem. Then mount it to **/mnt/data-lvm** using **/dev/vdc1** & **/dev/vdc2**.

Here are some details:

* Volume Group name = vg0
* Size of /dev/vdc1 = 1G
* Size of /dev/vdc2=1G

|  |
| --- |
|  |

1. parted /dev/vdc mkpart primary 1mb 1.1gb
2. parted /dev/vdc mkpart primary 1.1gb 2.2gb
3. parted /dev/vdc set 1 lvm on
4. parted /dev/vdc set 2 lvm on
5. vgcreate vg0 /dev/vdc1 /dev/vdc2
6. lvcreate -n lv0 -L 500mb vg0
7. mkfs.xfs /dev/vg0/lv0
8. mkdir /mnt/data-lvm
9. vi /etc/fstab
10. /dev/vg0/lv0 /mnt/data-lvm xfs defaults 1 2
11. mount /mnt/data-lvm
12. lsblk –fs

**DESCRIPTION**

**Create Physical Volume:** sudo pvcreate /dev/sdXY

**Create Volume Group:** sudo vgcreate vg\_name /dev/sdXY

**Volume Group Information:** sudo vgdisplay vg\_name

**Extend Volume Group:** sudo vgextend vg\_name /dev/sdXZ

**Reduce Volume Group:** sudo vgreduce vg\_name /dev/sdXZ

**Create Logical Volume:** sudo lvcreate -L sizeG -n lv\_name vg\_name

**Extend Logical Volume:** sudo lvextend -L +sizeG /dev/vg\_name/lv\_name

**Resize the Filesystem on Logical Volume:** sudo resize2fs /dev/vg\_name/lv\_name

**Remove Logical Volume:** sudo lvremove /dev/vg\_name/lv\_name

**Remove Volume Group:** sudo vgremove vg\_name

**Remove Physical Volume:** sudo pvremove /dev/sdXY

**Task 08 - Implementing Advanced Storage Feature**

“Extend existing logical volume **lv0** from **500MB** to **700MB** and mount to **/mnt/data-lvm**.”

1. lvextend -L +200M /dev/vg0/lv0 **(hanya ini)**
2. xfs\_growfs /dev/vg0/lv0
3. mount /dev/vg0/lv0 /mnt/data-lvm

**Task 09 - Implementing Advanced Storage Feature**

On **servera**, create stratis pool **str0** and filesystem **str0fs0** from **/dev/vdd**

* mount **str0fs0** to **/mnt/stratis**
* create snapshot of **str0fs0** with name **str0fs0-snap** and mount to **/mnt/stratis-snap**

1. systemctl status stratisd.service
2. stratis pool create str0 /dev/vdd
3. stratis filesystem create str0 str0fs0
4. stratis filesystem snapshot str0 str0fs0 str0fs0-snap
5. stratis filesystem list
6. mkdir /mnt/stratis
7. mount /stratis/str0/str0fs0 /mnt/stratis/
8. mkdir /mnt/stratis-snap
9. mount /stratis/str0/str0fs0-snap /mnt/stratis-snap/

**DESCRIPTION**

**Create a Pool:** sudo stratis pool create pool\_name /dev/sdXY

**Add Devices to a Pool:** sudo stratis pool add-data pool\_name /dev/sdXZ

**List All Pools:** sudo stratis pool list

**Show Pool Details:** sudo stratis pool show pool\_name

**Create a Filesystem:** sudo stratis filesystem create pool\_name fs\_name

**Create a Snapshot:** sudo stratis filesystem snapshot pool\_name fs\_name snapshot\_name

**List Snapshots:** sudo stratis filesystem list-snapshots pool\_name fs\_name

**Destroy a Snapshot:** sudo stratis filesystem destroy-snapshot pool\_name fs\_name snapshot\_name

**Resize a Filesystem:** sudo stratis filesystem destroy-snapshot pool\_name fs\_name snapshot\_name

**Destroy a Filesystem:** sudo stratis filesystem destroy pool\_name fs\_name

**Destroy a Pool:** sudo stratis pool destroy pool\_name

**Task 10 - Implementing Advanced Storage Feature**

On **serverb**, create the VDO volume **vdo1**, using the **/dev/vdb** device. Set its logical size to **50 GB**.

Format the **vdo1** volume with the XFS file-system type and mount it on **/mnt/vdo1**.

1. vdo create --name=vdo1 --device=/dev/vdb --vdoLogicalSize=50G
2. mkfs.xfs /dev/mapper/vdo1
3. mkdir /mnt/vdo1
4. mount /dev/mapper/vdo1 /mnt/vdo1/

**DESCRIPTION**

**Installing VDO:** sudo yum install vdo

**Create a VDO:** sudo vdo create --name=vdo\_volume\_name --device=/dev/sdXY --vdoLogicalSize=size --vdoSlabSize=size

**Start VDO:** sudo vdo start --name=vdo\_volume\_name

**Stop VDO:** sudo vdo stop --name=vdo\_volume\_name

**List VDO:** sudo vdo list

**VDO Volume Information:** sudo vdo status --name=vdo\_volume\_name

**Resize VDO:** sudo vdo grow --name=vdo\_volume\_name --vdoLogicalSize=new\_size

**Task 11 - Access Network-attached Storage**

An IT support company uses a central server, serverb, to host some shared directories on **/shares** for their groups and users.

Users need to be able to log in and have their shared directories mounted on demand and ready to use, under the **/shares** directory on servera.

Important information:

* **serverb** is sharing the **/shares** directory, which in turn contains the management, production and operation subdirectories.
* The **managers** group consists of the **manager1** and **manager2** users. They have read and write access to the **/shares/management** shared directory.
* The **production** group consists of the **dbuser1** and **sysadmin1** users. They have read and write access to the **/shares/production** shared directory.
* The **operators** group consists of the **consultant1** users. They have read and write access to the **/shares/operation** shared directory.
* The main mount point for **servera** is the **/remote** directory
* Use the **/etc/auto.master.d/shares.autofs** file as the master map file and the **/etc/auto.shares** file as the indirect map file.
* The **/shares/management** exported directory is automounted on **/remote/management** on **servera**.
* The **/shares/production** exported directory is automounted on **/remote/production** on **servera**.
* The **/shares/operation** exported directory is automounted on **/remote/operation** on **servera**.

1. yum install autofs
2. sed -i 's/192.168.0.254/202.65.116.10/g' /etc/hosts **(optional)**
3. vi /etc/auto.master.d/shares.autofs

/remote /etc/auto.shares

1. vi /etc/auto.shares

management serverb:/shares/management

production serverb:/shares/production

operation serverb:/shares/operation

1. systemctl restart autofs
2. chmod 755 /remote/management /remote/production /remote/operation
3. ls /remote

**DESCRIPTION**

**Install Autofs:** sudo yum install autofs

**Configure Autofs (/etc/auto.master):** /mnt/nas /etc/auto.nas

**Create an Autofs Map:** share\_name -fstype=nfs server\_ip:/path/to/share

**Restart Autofs:** sudo systemctl restart autofs

**Task 12 - Scheduling Job**

On **servera**, configure a task : **run echo "hello" >> /home/natasha/file\_cron** command every **2 minutes** in user **natasha**.

1. crontab -u natasha -e
2. \*/2 \* \* \* \* echo "hello" >> /home/natasha/file\_cron
3. crontab -u natasha -l
4. cat /home/natasha/file\_cron

**DESCRIPTION**

**crontab [options] file**

**crontab [options]**

**crontab -n [hostname]**

**Options:**

**-u <user>** define user

**-e** edit user's crontab

**-l** list user's cr ontab

**-r** delete user's crontab

**-I** prompt before deleting

**-n <host>** set host in cluster to run users' crontabs

**-c** get host in cluster to run users' crontabs

**-s** selinux context

**-V** print version and exit

**-x <mask>** enable debugging

**CRONTAB FILE SYNTAX**

minute hour day\_of\_month month day\_of\_week command\_to\_run

* **minute** (0-59)
* **hour** (0-23)
* **day\_of\_month** (1-31)
* **month** (1-12)
* **day\_of\_week** (0-6, where 0 is Sunday)

**SPECIAL CHARACTERS**

* **(\*) :** The asterisk is a wildcard that represents "any value" for the respective time unit. Example:

**`\*` in the minute field means "every minute."**

* **(,) :**The comma is used to specify a list of values for a time unit. Example:

**`1,15,30` in the minute field means "at the 1st, 15th, and 30th minute."**

* **(-) :**The hyphen is used to specify a range of values for a time unit. Example:

**`10-15` in the hour field means "from 10 AM to 3 PM."**

* **(/) :**The slash is used to define a step value for a time unit. Example:

**`\*/5` in the minute field means "every 5 minutes."**

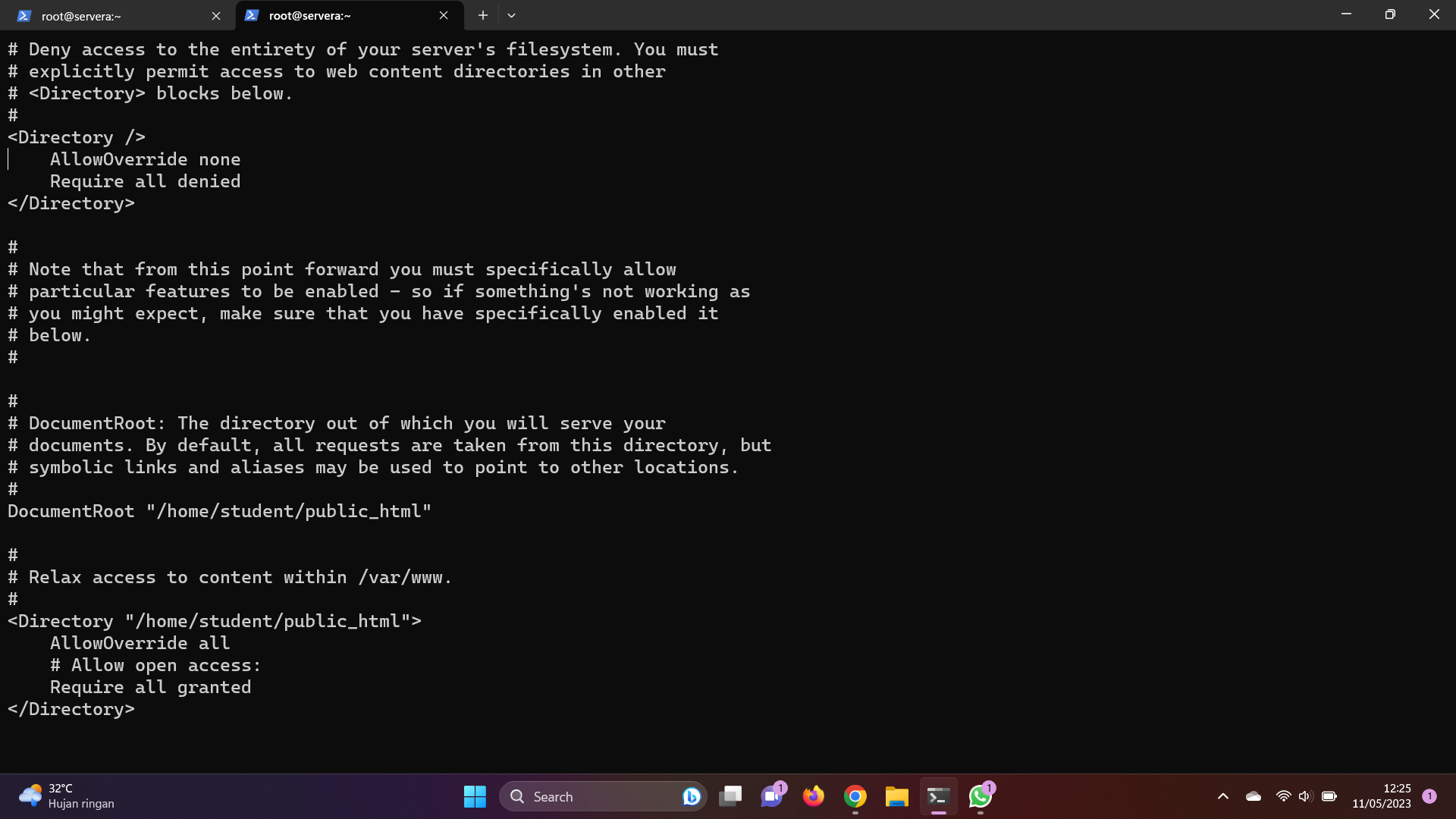
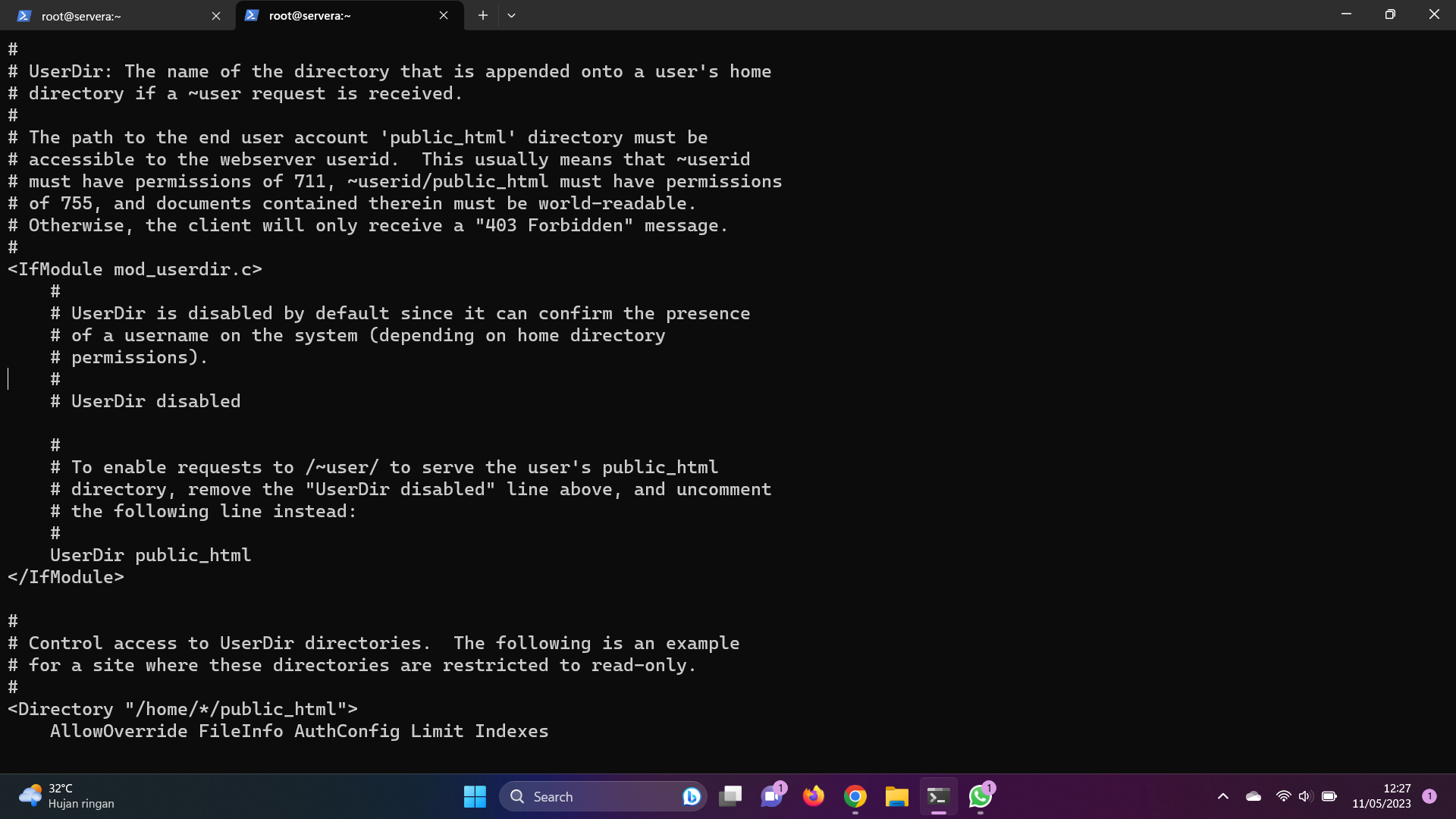
**Task 13 – SELinux**

**1.** Access to **http://servera/index.html**.  
**2.** You will see the error message: **index.html** cannot be accessed from the httpd service.  
**3.** Resolve the issue preventing Apache (httpd) from serving web content.  
**4.** Verify that Apache (httpd) is able to serve web content.

1. curl http://servera/index.html
2. systemctl status httpd
3. semanage fcontext -a -t httpd\_sys\_content\_t “/var/www/html(/.\*)?”
4. restorecon -R /var/www/html
5. curl <http://servera/index.html>

**Exam Task 14 – SELinux**

1. On servera, configure **httpd** to use a document root in a non-standard location to **/home/student/public\_html**.  
2. Verify that **Apache (httpd)** is able to serve web content. (**http://servera/~student/index.html**).

1. curl <http://servera/~student/index.html>
2. vi /etc/httpd/conf/httpd.conf **(ganti dokumen root)**
3. chmod 711 /home/student
4. setsebool -P httpd\_enable\_homedirs on
5. vi /etc/httpd/conf.d/userdir.conf **(tambah dan hilangkan tanda #)**
6. systemctl restart httpd
7. curl http://servera/~student/index.html

**Exam Task 15 – ACL**

On servera

**1.** Copy **/etc/fstab** to **/var/tmp**.  
**2.** change group ownership file **/var/tmp/fstab** to group **admin**.  
**3.** Set ACL for file **/var/tmp/fstab** the user natasha could read, write and execute it, while harry without any permission.

1. cp /etc/fstab /var/tmp
2. chgrp admin /var/tmp/fstab
3. setfacl -m u:natasha:rwx /var/tmp/fstab
4. setfacl -m u:harry:0 /var/tmp/fstab
5. getfacl /var/tmp/fstab

**Exam Task 16 – ACL**

On servera

**1.** Existing users : contractor1, contractor2, contractor3, member group contractors.  
**2.** Set ACL permission read, write, and execute to group contractors to file **/home/contractors/filecon**.  
**3.** Ser ACL permission read to user contractor2.

1. chgrp contractors /home/contractors/filecon
2. setfacl -m g:contractors:rwx /home/contractors/filecon
3. setfacl -m u:contractor2:r /home/contractors/filecon
4. getfacl /home/contractors/filecon

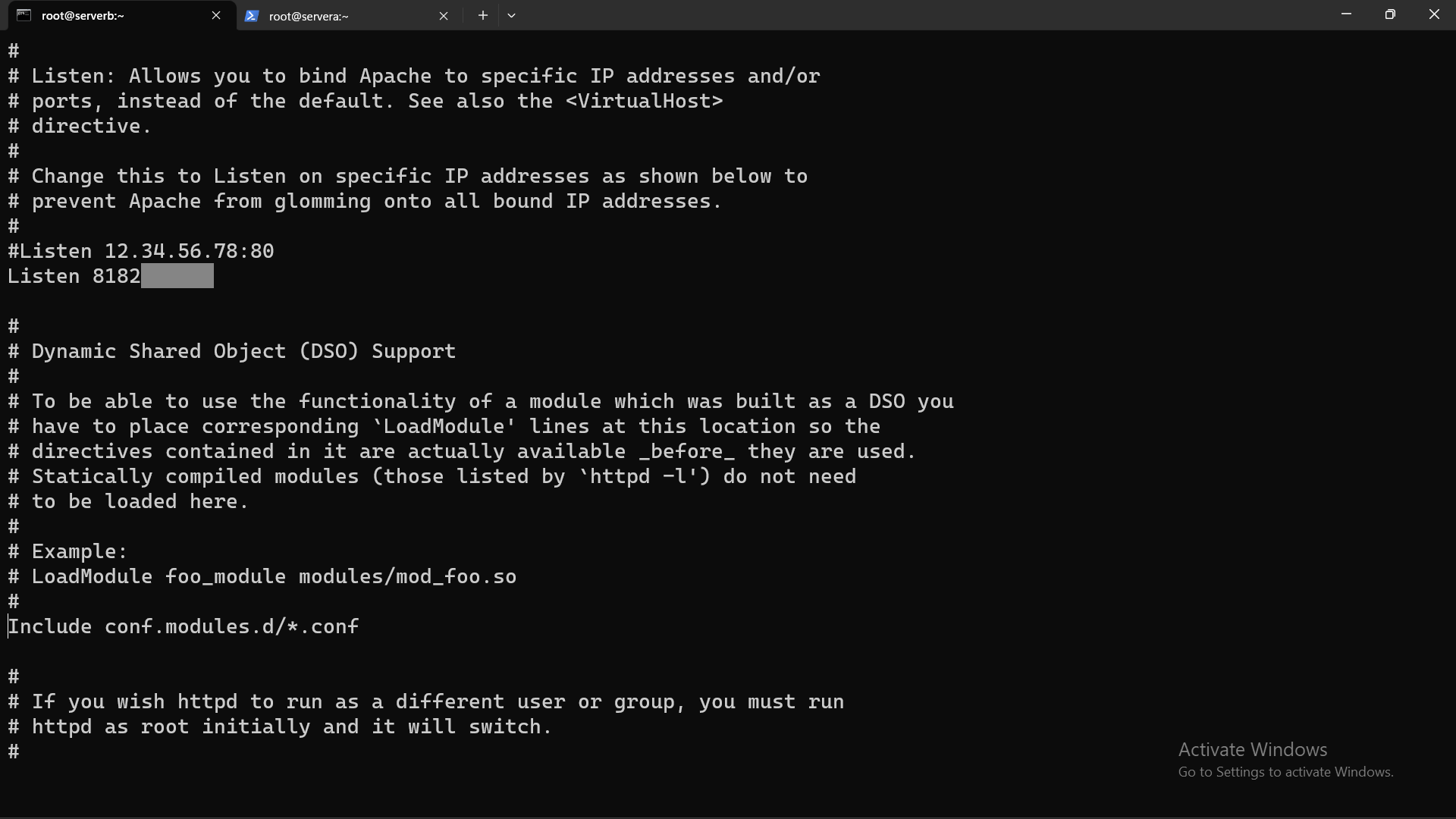
**Exam Task 17 - Network Security**

**1.** On **serverb**, configure **httpd** to use a non-standard http port on **8182/TCP**.  
**2.** Verify that Apache (**httpd**) is able to serve web content. (**http://serverb:8182/index.html**).

1. curl http://serverb:8182/index.html
2. semanage port -a -t http\_port\_t -p tcp 8182

semanage port --add --type http\_port\_t --proto tcp 8182 ***(opsi apabila sebelumnya gagal)***

1. firewall-cmd --permanent --add-port=8182/tcp
2. firewall-cmd --reload
3. vi /etc/httpd/conf/httpd.conf

Listen 80 -> 8182

1. systemctl restart httpd
2. curl http://serverb:8182/index.html

**Exam Task 18 - Linux Link**

On servera

1. Create a hard link named **/home/student/fileA-hard-link.txt** for the existing file, **/home/student/fileA** on **servera**.  
2. Create a soft link named **/home/student/fileB-soft-link.txt** that points to the for the existing file, **/home/student/fileB**.

1. ln /home/student/fileA /home/student/fileA-hard-link.txt
2. ln -s /home/student/fileB /home/student/fileB-soft-link.txt
3. ls -l /home/student

**Exam Task 19 - Running Containers**

*Note: disarankan untuk menggunakan lebih dari 1 tab terminal cara dibawah menggunakan 1 tab terminal*

**1.** Access **podsvc** user on serverb via ssh. The password for the **podsvc** user is **redhat**. Change **podsvc** user password to **podsvc\_pass**.

**2.** Login to image registry **registry.lab.example.com** use the **admin** account with **redhat321** for the password.

**3.** Pull image **httpd-24:1-105** image from registry **registry.lab.example.com/rhel8/**.

**4.** Create the **/home/podsvc/srv/web** directory. Prepare the directory so that containers have read/write access. You will use this directory for persistent storage.

**5.** Create index.html file using the following command

echo "Hello World" > /home/podsvc/srv/web/index.html

**6.** On serverb, as the **podsvc** user, create a detached webserver container named **web** . Use the **httpd-24:1-105** image from the **registry.lab.example.com**. Mount the **/home/podsvc/srv/web** directory on the host as **/var/www/html** and redirect port **8888** on the local host to the container port **8080**.

**7.** On serverb, as the **podsvc** user, configure systemd so that the **web** container starts automatically with the server.

**On root serverb**

1. passwd podsvc
2. masukan password: podsvc\_pass

masukan kembali password

1. sed -i 's/192.168.0.200/202.65.116.10/g' /etc/hosts **(jika diperlukan)**
2. visudo/vim /etc/sudoers

podsvc ALL=(ALL) ALL

1. mkdir /home/podsvc/srv/web
2. semanage fcontext -a -t container\_file\_t "/home/podsvc/srv(/.\*)?"
3. restorecon -R /home/podsvc/srv

chcon -Rv --type=httpd\_sys\_content\_t /home/podsvc/srv/web (jika command diatas error)

1. echo "Hello World" > /home/podsvc/srv/web/index.html

**On podsvc**

1. podman login registry.lab.example.com --tls-verify=false

Username: admin

Password: redhat321

1. podman pull registry.lab.example.com/rhel8/httpd-24:1-105 --tls-verify=false
2. podman run -d --name web -p 8888:8080 -v /home/podsvc/srv/web:/var/www/html registry.lab.example.com/rhel8/httpd-24:1-105
3. podman ps
4. curl <http://localhost:8888>

**Other Materials:**

**Viewing File Permissions:**

To view the permissions of a file or directory, use the **`ls -l`** command. It displays detailed information, including the permissions, for each file or directory listed.

**Understanding Permission Notations:**

File permissions are represented using a combination of letters and symbols. The permissions are divided into three sets:

* Owner permissions: The permissions for the file/directory owner.
* Group permissions: The permissions for the file/directory group.
* Other permissions: The permissions for everyone else.

**The permission notations are as follows:**

* **`r`** (read): Allows reading/viewing a file or listing files in a directory.
* **`w`** (write): Allows modifying a file or adding/removing files in a directory.
* **`x`** (execute): Allows executing or running a file (for directories, allows accessing files within).
* **`-`** (hyphen): Indicates that a specific permission is not granted.

**Changing File Permissions:**

* To change file permissions, use the **`chmod`** command followed by the permission notation and the file/directory name.
* For example, to give read and write permissions to the owner of a file, use the following command:

chmod u+rw file.txt

* Similarly, you can use **`g`** for group permissions and **`o`** for other permissions. Use **`a`** (all) to set permissions for all sets.
* Numeric values can also be used to set permissions. Each permission has a corresponding numeric value:
* `r` (read): 4
* `w` (write): 2
* `x` (execute): 1
* Add the numeric values of the desired permissions to set permissions. For example, to give read and write permissions to the owner and group, you can use:

chmod 660 file.txt

**Recursively Changing Permissions:**

* To change permissions recursively for a directory and its contents, use the **`-R`** option with the **`chmod`** command.
* For example, to give read and write permissions to the owner and group recursively, use:

chmod -R u+rw,g+rw directory/

**Changing Ownership:**

* To change the ownership of a file or directory, use the `chown` command followed by the new owner and the file/directory name.
* For example, to change the owner of a file to a specific user, use:

chown new\_owner file.txt

* You can also change the group ownership using the `chgrp` command.

**Default File Permissions:**

* Newly created files and directories inherit permissions based on the system's umask value and the default settings of the file creation utility.
* The umask value determines which permissions are subtracted from the default permissions.
* You can modify the umask value using the **`umask`** command or by editing the shell configuration file.

**Special Permissions:**

* Set UID **(u+s)**
* The setuid permission is denoted by an **"s"** in the **user** execute position (the owner's execute permission).
* When set on an executable file, it allows the user who executes the file to temporarily assume the privileges of the file owner.
* It is commonly used to allow users to execute certain programs with elevated privileges.

chmod u+s executable\_file

* Set GID **(g+s)**
* The setgid permission is denoted by an **"s"** in the **group** execute position (the group's execute permission).
* When set on a directory, it allows files created within that directory to inherit the group ownership of the parent directory instead of the user's default group.
* It is commonly used to ensure collaborative file sharing within a group.

chmod g+s directory

* Sticky Bit **(t)**
* The sticky bit permission is denoted by a **"t"** in the other execute position (execute permission for others).
* When set on a directory, it restricts the deletion or renaming of files within that directory to the file owner, the directory owner, or the root user.
* It is commonly used on directories where multiple users need write access, such as temporary directories.

chmod +t directory

**Managing Compressed tar Archives Objectives:**

**Overview of tar Operations:**

| **Option** | **Description** |
| --- | --- |
| -c, --create | Create a new archive. |
| -x, --extract | Extract from an existing archive. |
| -t, --list | List the table of contents of an archive. |

**Selected tar General Options:**

| **Option** | **Description** |
| --- | --- |
| -v, --verbose | Verbose. Shows which files get archived or extracted. |
| -f, --file= | File name. This option must be followed by the file name of the archive to use or create. |
| -p, --preserve-permissions | Preserve the permissions of files and directories when extracting an archive, without subtracting the umask. |

**Overview of tar Compression Options:**

| **Option** | **Description** |
| --- | --- |
| -z, --gzip | Use gzip compression (.tar.gz). |
| -j, --bzip2 | Use bzip2 compression (.tar.bz2). bzip2 typically achieves a better compression ratio than gzip. |
| -J, --xz | Use xz compression (.tar.xz). The xz compression typically achieves a better compression ratio than bzip2. |

**Listing Options of the tar Command**

The **tar** command expects one of the three following options:

* Use the **-c** or --create option to create an archive.
* Use the **-t** or --list option to list the contents of an archive.
* Use the **-x** or --extract option to extract an archive.

Other commonly used options are:

* Use the **-f** or --file= option with a file name as an argument of the archive to operate.
* Use the **-v**or --verbose option for verbosity; useful to see which files get added to or extracted from the archive.

The following command creates an archive named archive.tar with the contents of file1, file2, and file3 in the user's home directory.

[user@host ~]$ tar **-cf** archive.tar file1 file2 file3

[user@host ~]$ ls archive.tar

archive.tar

Good Luck Four Your Exam