Operating Systems
Constructor University
Dr. Jürgen Schönwälder

Module: CO-562
Date: 2026-11-20
Due: 2026-11-27

Problem Sheet #12

Problem 12.1: *file system permissions* (1+1 = 2 points)

Unix file system objects have basic permissions associated with (i) the file owner, (ii) the file's group members, and (iii) everybody else with access to the file system. Answer the following questions:

a) What is the meaning of the following file access permissions?

```
$ ls -l foo
-rwxrw-r-- 1 alice co-562 0 Nov 30 14:53 foo
$ ls -ld bar
drwx-wx--- 2 alice co-562 4096 Nov 30 14:56 bar
$ ls -l /usr/bin/sudo
-rwsr-xr-x 1 root root 182600 Feb 27  2021 /usr/bin/sudo
$ ls -ld /tmp
drwxrwxrwt 8 root root 700 Nov 17 13:02 /tmp
$ ls -ld /usr/local/bin
drwxrwsr-x 2 root staff 4096 Jul 30  2018 /usr/local/bin
```

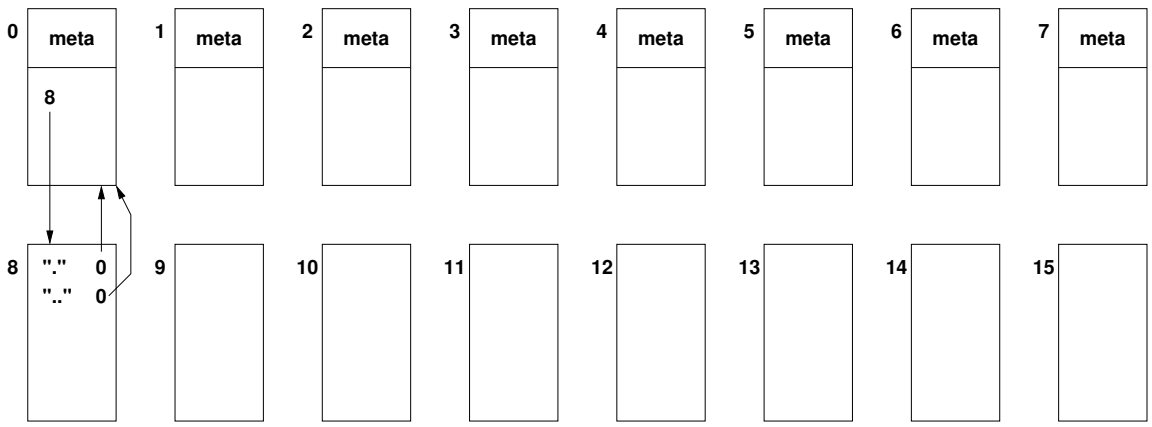
b) Can the regular user `bob` (with a umask of 0022), a member of the group `co-562`, read the content of the directory `bar`? Can `bob` create a file in the directory `bar`? Explain.

Bob executes the following shell commands. What are the file permissions of the new file and who is the owner of the file? Explain.

```
$ rm -f world
$ sudo echo hello > world
```

Problem 12.2: *file system updates* (3 points)

Consider a small file system residing on a block device with 16 blocks (numbered 0..15). All blocks have the same size. The first 8 blocks (0...7) are reserved for inodes. An inode stores file attributes (metadata) of a file system object as well as references to data blocks. The remaining blocks (8...15) are reserved for storing data or directory information, they are called data blocks, or short dnodes. The root directory is always found in inode 0. Below is a figure visualizing the situation right after initializing the file system.



In the following, we focus on the block references that the file system maintains. The initial situation can be described as follows:

```
inode 0: 8                      // inode 0 refers to dnode(s) 8
inode i: undef                  // inode i has undefined content (i in {1..7})
dnode 8: {(".", 0), ("..", 0)}  // dnode 8 has 2 directory entries (to inode 0)
dnode i: undef                  // dnode i has undefined content (i in {9..15})
freeblks: 0111111101111111      // free blocks bitmap (0 = used, 1 = free)
```

A sequence of changes is made to the file system. In the following steps, write down (using the notation shown above) which inodes and/or dnodes are updated and how the freeblks bitmap changes. In each step, write down only the changes. If you need to allocate unused inodes or dnodes, choose lower numbered unused inodes or dnodes first.

a) A directory `/a` is created in the root directory. `mkdir /a`

b) A directory `/a/b` is created in the directory `/a`. `mkdir /a/b`

c) A file `c` is created in `/a/b` with the content `hello`. `echo "hello" > /a/b/c`

d) The file `/a/b/c` is moved into the directory `/a`. `mv /a/b/c /a`

e) A link is created such that `/a/c` is also known as `/a/b/d`. `ln /a/c /a/b/d`

f) A symbolic link is created resolving `/e` to `/a/b/d`. `ln -s /a/b/d /e`

Problem 12.3: *logical volume management* (1+1+1+1+1 = 5 points)

On Linux, you can create block storage devices using regular files as the backend storage. This is useful for experimentation and debugging. You will use this to create physical and logical volumes.

You may want to do this exercise on a virtual machine to make sure you avoid any catastrophic results by accidentally mistyping a command.

Creating block devices on top of regular files is accomplished by loop devices. First, you have to load to the loop kernel module:

```
sudo modprobe loop
```

Next, you create 5 storage devices, each having a capacity of $20\,\mathrm{MB}$.

```
for i in $(seq 0 4) ; do
    img=loop$i.img
    dev=/dev/loop$i
    dd if=/dev/zero of=$img bs=1M count=20
    sudo losetup $dev $img
done
```

To list your loop devices, you can use this command:

```
sudo losetup -a
```

To remove all loop devices and the files, you can run this loop:

```
 for i in $(seq 0 4) ; do
    img=loop$i.img
    dev=/dev/loop$i
    sudo losetup -d $dev
    rm $img
done
```

In order to create logical volumes, you need to install the `lvm2` package (if not installed yet). On Debian or Ubuntu systems, the package can be installed by using the following command:

```
sudo apt-get install lvm2
```

You can get an overview over the lvm commands in the `lvm` man page in section 8 of the Unix manual. Every individual lvm command has its own man page in section 8 as well.

Carry out the following experiments and provide the requested information:

a) Create a volume group named `vg0` that consists of the three physical volumes `/dev/loop0`, `/dev/loop1`, and `/dev/loop2`. Document the commands that you have used and show the output of the commands `sudo pvs` and `sudo vgs`. Look at the `PSize`, what do you observe?

b) Within volume group `vg0`, create a logical volume `lv0` with a size of $20\,\mathrm{MB}$. Create a file system in `lv0` and mount it on `/mnt`. Show the output of the commands `sudo lvs`, `sudo pvs`, and `sudo df /mnt`.

c) You need to grow the file system on `lv0` to about $60\,\mathrm{MB}$. What are the commands that you use and why? Show the the output of the commands `sudo lvs`, `sudo pvs`, and `sudo df /mnt`.

d) Create a snapshot logical volume called `lv0s` of `lv0`. Show the output of the commands `sudo pvs` and `sudo lvs`. Write a short message into the file in `/mnt/msg.txt` (assuming `lv0` is still mounted on `/mnt`). Show the output of the command `sudo lvs` again. What has changed and why? Mount the snapshot volume and verify that the file does not exist in the snapshot file system.

e) Remove all logical volumes from `vg0` and make sure all loop devices have been added as physical volumes to `vg0`. Create a RAID 1 logical volume `lv0r1` with the size of 12Mb. Which physical volumes are used to store the data of `lv0r1`? Create a RAID 5 logical volume `lv1r5` using all the remaining free physical extents. Show the output of the command `sudo lvs`. What is the size of `lv1r5` and what is the size of the physical volumes providing storage for `lv1r5`?