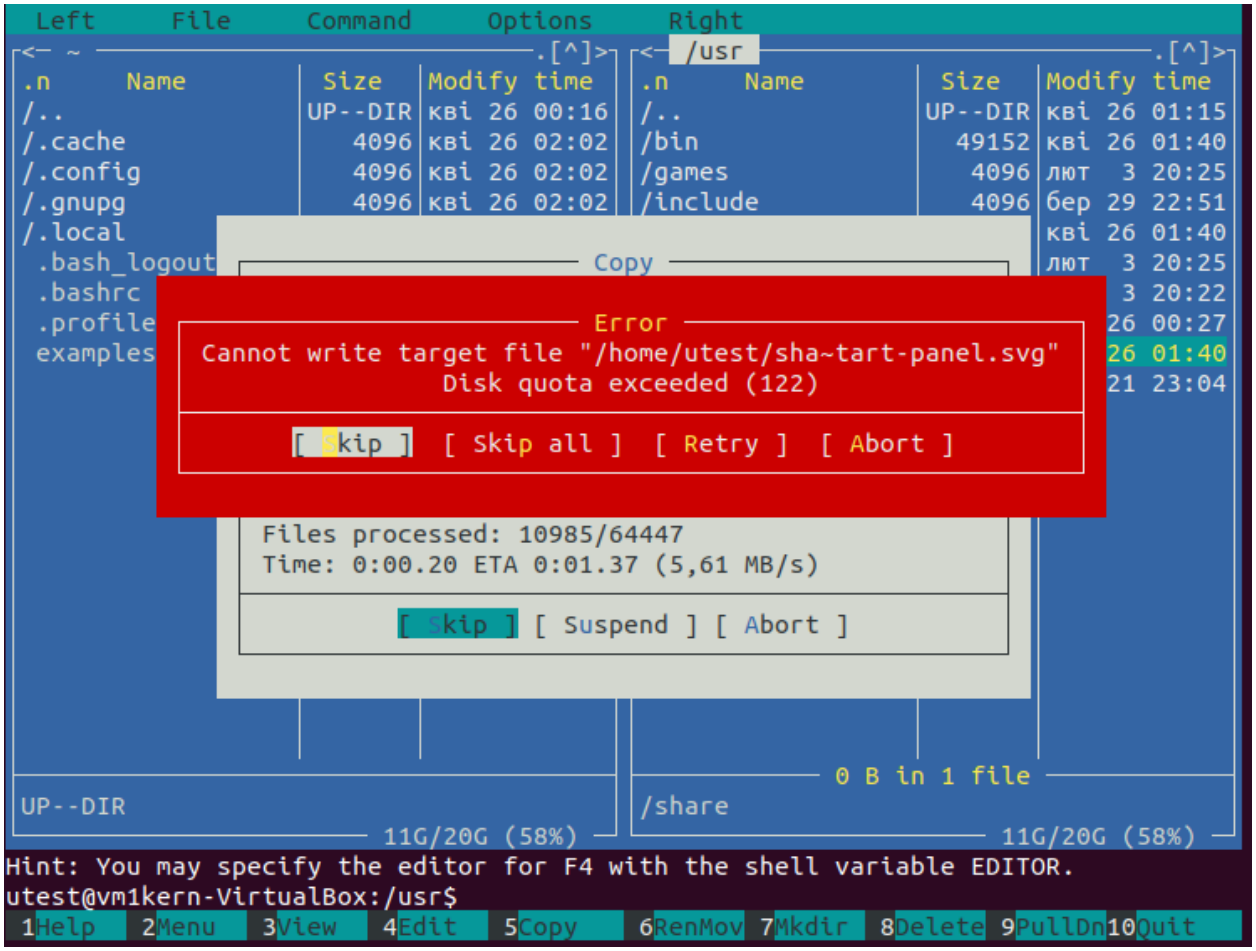
EPAM University Programs
DevOps external course
Module 4 Linux & Bash Essentials
TASK 4.7

Part1. **Quota allocation mechanism.**

Employing commands from presentation #4.6, create a new user, say, *utest*. Based on the quota mechanism, limit the available disk space for this user to **soft**: 100M and **hard**: 150M.
Then, using Midnight Commander (since MC shows warnings about exceeding the limits of available to a user disk space), copy content of /usr directory to utest's home directory (actually, /usr isn't mandatory, you are free to copy any other data, the only condition is sufficient total size of the files to copy).
Note: if /home is not a mount point, then the **mount** and **quotaon** commands should be called with respect to the root partition /.
Note 2: Please, put into your report screenshots of your terminal window with the executed commands, along with screenshots of MC panels over which quota warnings are shown (i.e. warnings about exceeding soft and hard limits).

```
*** Report for user quotas on device /dev/sda1
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      -- 7207456       0       0         217564     0     0
man       --    1256       0       0             83     0     0
systemd-network --      12       0       0              3     0     0
syslog    --    2328       0       0             14     0     0
_apt      --      28       0       0              4     0     0
avahi-autoipd --       4       0       0              1     0     0
dnsmasq   --       4       0       0              1     0     0
speech-dispatcher --       8       0       0              2     0     0
colord    --      56       0       0              5     0     0
hplip     --       4       0       0              1     0     0
geoclue   --       8       0       0              2     0     0
gdm       --     232       0       0             41     0     0
vm1kern   --   63420       0       0            811     0     0
utest     +-  150000  100000  150000  6days   17946     0     0
#62583    --       4       0       0              2     0     0
```



Part2. **Access Control Lists, ACLs**

In what follows, we assume that there are two users: *guest* (included into the list of sudoers) and *utest*. None of the users is the superuser (i.e. UIDs of the users differ from 0).

The most task: to allow user *utest* visit *guest*'s home directory.

The average task: to acquaint yourself with the basics of ACL and verify the fact that ACL privileges override the **chmod** ones.

Before proceeding to the task execution, please, visit the linux.org page describing ACL, https://linuxconfig.org/how-to-manage-acls-on-linux.

Every step of execution should be stored into some file **/var/log** directory (use logger, please).

1. Based on given in presentation #4.7 instructions, turn on and set up the ACL. *Caution*! The fact that a file system has been mounted with the "acl" flag on by default, doesn't mean that the ACL package is installed.

Prior to any action, it is advised to check if the “acl” flag is on, using
tune2fs -l /dev/sda*
(a particular name of the device file sda*, is to be determined by calling to **blkid**, invoke it twice:
(i) on behalf of *guest* (i.e. without the superuser privileges);

```
vm1kern@vm1kern-VirtualBox:~$ blkid
/dev/sr0: UUID="2020-02-18-17-20-05-35" LABEL="VBox_GAs_6.1.4" TYPE="iso
9660"
```

(ii) with **sudo** (i.e. with the superuser privileges). Note the level of details provided by different **blkid** outputs).

```
vm1kern@vm1kern-VirtualBox:~$ sudo blkid
/dev/loop0: TYPE="squashfs"
/dev/loop1: TYPE="squashfs"
/dev/loop2: TYPE="squashfs"
/dev/loop3: TYPE="squashfs"
/dev/loop4: TYPE="squashfs"
/dev/loop5: TYPE="squashfs"
/dev/loop6: TYPE="squashfs"
/dev/sr0: UUID="2020-02-18-17-20-05-35" LABEL="VBox_GAs_6.1.4" TYPE="iso
9660"
/dev/sda1: UUID="74f0345e-93ca-4bb9-9de8-40d28df9b873" TYPE="ext4" PARTU
UID="b5210a9d-01"
/dev/loop8: TYPE="squashfs"
```

2. Log in as *guest*. Create in */tmp* a directory called *acl_test*. By means of **chmod,** allow user utest to perform all possible operations (rwx) with respect to *acl_test.* Verify that user *utest* is indeed capable of implementing granted him (her) privileges. For example, acer logging in as *utest*, create a file in */tmp/acl_test*, say, *utest.txt* with the aid of **touch**. Query information about the directory and file by calling to
ls -ld /tmp/acl_test

```
vm1kern@vm1kern-VirtualBox:/tmp$ su utest
Password:
$ cd /test/acl_test
sh: 1: cd: can't cd to /test/acl_test
$ touch /tmp/acl_test/utest.txt
$ ls -ld /tmp/acl_test
drwxrwxrwx 2 vm1kern vm1kern 4096 кві 28 02:58 /tmp/acl_test
```

ls -l /tmp/acl_test

```
$ ls -l /tmp/acl_test
total 0
-rw-rw-r-- 1 utest utest 0 кві 28 02:58 utest.txt
```

To check ACL permissions do:

ge4acl /tmp/acl_test

```
$ getfacl /tmp/acl_test
getfacl: Removing leading '/' from absolute path names
# file: tmp/acl_test
# owner: vm1kern
# group: vm1kern
user::rwx
group::rwx
other::rwx
```

ge4acl /tmp/acl_test/utest.txt

```
$ getfacl /tmp/acl_test/utest.txt
getfacl: Removing leading '/' from absolute path names
# file: tmp/acl_test/utest.txt
# owner: utest
# group: utest
user::rw-
group::rw-
other::r--
```

3. Employ ACL to block any activity except for reading, for user *utest* with respect to directory */tmp/acl_test*(hint: use **se4acl**). Test if the actions are effectively prohibited

touch /tmp/acl_test/prohibited.txt

Is it possible to invoke this command?

echo "new content" > /tmp/acl_test/utest.txt

Test if user *utest* can be prevented from modifying content of the file *utest.txt* by means of ACL. (Note that user *utest* is the owner of the file *tmp/acl_test/utest.txt*).

4. Consider a situation when at the ACL level user *utest* is allowed to have all possible privileges with respect to */tmp/acl_test*, while no ac=on is allowed with **chmod** (conventional mechanism). (Hint: repeat step 3, but given the new context).

5. For user *utest*, set default ACLs to the directory */tmp/acl_test* which allow read-only access (hint: use the -d option of the **se4acl** command). Being logged in as *utest*, invoke **touch** to create the file *utest2.txt* in the */tmp/acl_test* directory. Query permissions on this file using **ge4acl**.

6. Set the maximum permissions mask on the */tmp/acl_test/utest.txt* file in such a way as to allow read-only access. Check permissions with **ge4acl**.
7. Delete all ACL entries relative to the */tmp/acl_test* directory.