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).

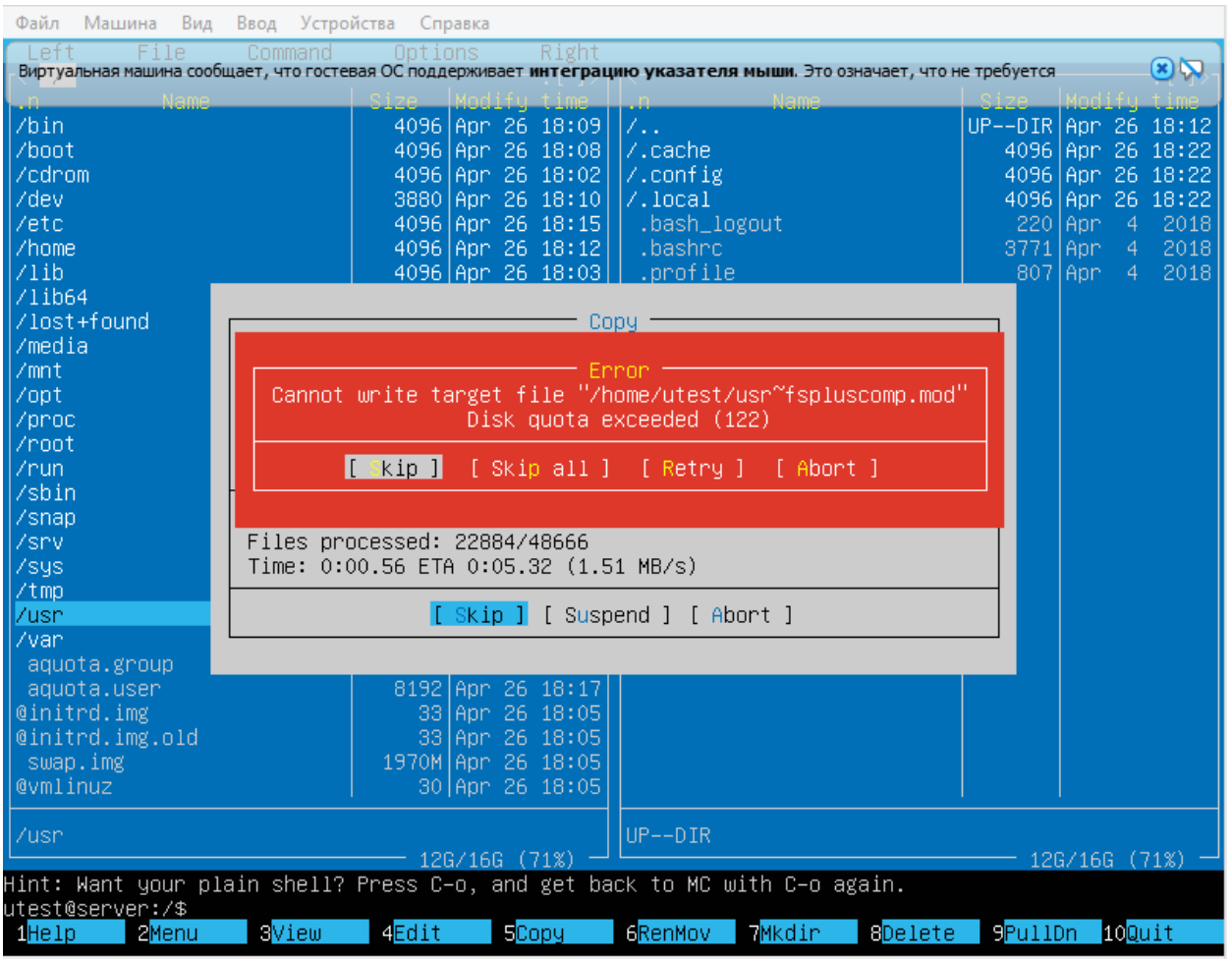
```
root@server:~# groupadd utest
root@server:~# useradd -g utest -s /bin/bash -d /home/utest -m utest
root@server:~# passwd utest
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

```
 GNU nano 2.9.3                          /tmp//EdP.a5VPL1Y

Disk quotas for user utest (uid 1001):
  Filesystem                blocks       soft       hard     inodes     soft     hard
  /dev/sda2                 153600     102400     153600      29685        0        0
```

```
alex@server:~$ sudo quota -vs utest
Disk quotas for user utest (uid 1001):
     Filesystem   space   quota   limit   grace   files   quota   limit   grace
      /dev/sda2    150M*   100M    150M   6days   29685       0       0
```

```
alex@server:~$ sudo quota -vs utest
Disk quotas for user utest (uid 1001):
     Filesystem   space   quota   limit   grace   files   quota   limit   grace
      /dev/sda2    150M*   100M    150M   6days   29685       0       0
alex@server:~$ sudo repquota -s /
*** Report for user quotas on device /dev/sda2
Block grace time: 7days; Inode grace time: 7days
                        Space limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root      --   3871M      0K      0K          72235     0     0
daemon    --     64K      0K      0K              4     0     0
man       --   1276K      0K      0K            141     0     0
systemd-network --     12K      0K      0K              3     0     0
syslog    --    236K      0K      0K              5     0     0
_apt      --     24K      0K      0K              4     0     0
lxd       --      4K      0K      0K              1     0     0
dnsmasq   --      4K      0K      0K              1     0     0
landscape --      8K      0K      0K              3     0     0
pollinate --      4K      0K      0K              2     0     0
alex      --     28K      0K      0K              9     0     0
utest     +-    150M    100M    150M  6days    29685     0     0
#62583    --      4K      0K      0K              2     0     0
```



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).

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).

```
alex@server:~$ ls /home/
alex  utest
alex@server:~$ groupadd guest
groupadd: Permission denied.
groupadd: cannot lock /etc/group; try again later.
alex@server:~$ sudo groupadd guest
[sudo] password for alex:
alex@server:~$ useradd -g guest -s /bin/bash -d /home/guest -m guest
useradd: Permission denied.
useradd: cannot lock /etc/passwd; try again later.
alex@server:~$ sudo useradd -g guest -s /bin/bash -d /home/guest -m guest
alex@server:~$ sudo passwd guest
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
alex@server:~$ _
```

```
alex:x:1000:1000:alex:/home/alex:/bin/bash
utest:x:1001:1001::/home/utest:/bin/bash
guest:x:1002:1002::/home/guest:/bin/bash
```

```
utest@server:/home/alex$ visudo -f /etc/sudoers
```

```
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/b

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
test    ALL=(ALL:ALL) ALL
# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

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

```
alex@server:~$ sudo groupadd testing
[sudo] password for alex:
alex@server:~$ sudo usermod -aG testing utest
alex@server:~$ sudo usermod -aG testing guest
```

```
alex@server:~$ members testing
utest guest
alex@server:~$ sudo chgrp testing /home/guest
alex@server:~$ ls -l /home/ | grep guest
drwxr-xr-x 3 guest testing 4096 Apr 27 14:55 guest
alex@server:~$ sudo chmod o-rx /home/guest/
alex@server:~$ ls -l /home/ | grep guest
drwxr-x--- 3 guest testing 4096 Apr 27 14:55 guest
alex@server:~$ cd /home/guest
bash: cd: /home/guest: Permission denied
alex@server:~$
```

```
alex@server:~$ su utest
Password:
utest@server:/home/alex$ cd /home/guest
utest@server:/home/guest$ _
```

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*

```
  GNU nano 2.9.3                          /etc/fstab

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/a4738d87-6d63-47f6-892b-3bf22f669b09 / ext4 usrquota,grpquota,acl 0 0
/swap.img       none    swap    sw      0       0
```

(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);

```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
```

```
guest@server:~$ blkid
/dev/sda2: UUID="a4738d87-6d63-47f6-892b-3bf22f669b09" TYPE="ext4" PARTUUID="21f12a98-94e4-4c51-964b
-cb4c331b8113"
guest@server:~$
```

```
guest@server:~$ blkid | logger -t 1a
Apr 27 18:30:49 server 1a: /dev/sda2: UUID="a4738d87-6d63-47f6-892b-3bf22f669b09" TYPE="ext4" PARTUU
ID="21f12a98-94e4-4c51-964b-cb4c331b8113"
Apr 27 18:30:49 server 1a: /dev/loop0: TYPE="squashfs"
Apr 27 18:30:49 server 1a: /dev/loop1: TYPE="squashfs"
```

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

```
guest@server:~$ sudo blkid
[sudo] password for guest:
Sorry, try again.
[sudo] password for guest:
/dev/sda2: UUID="a4738d87-6d63-47f6-892b-3bf22f669b09" TYPE="ext4" PARTUUID="21f12a98-94e4-4c51-964b
-cb4c331b8113"
/dev/loop0: TYPE="squashfs"
/dev/loop1: TYPE="squashfs"
/dev/sda1: PARTUUID="5a6121ff-60b0-434c-8231-22caf8665176"
```

```
guest@server:~$ sudo blkid | logger -t 1b
Apr 27 18:31:00 server 1b: /dev/sda2: UUID="a4738d87-6d63-47f6-892b-3bf22f669b09" TYPE="ext4" PARTUU
ID="21f12a98-94e4-4c51-964b-cb4c331b8113"
Apr 27 18:31:00 server 1b: /dev/loop0: TYPE="squashfs"
Apr 27 18:31:00 server 1b: /dev/loop1: TYPE="squashfs"
Apr 27 18:31:00 server 1b: /dev/sda1: PARTUUID="5a6121ff-60b0-434c-8231-22caf8665176"
```

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

ls -l /tmp/acl_test

To check ACL permissions do:

getfacl /tmp/acl_test

getfacl /tmp/acl_test/utest.txt

```
guest@server:/root$ cd
guest@server:~$ mkdir /tmp/acl_test
guest@server:~$ chmod 777 /tmp/acl_test
guest@server:~$ su utest
Password:
utest@server:/home/guest$ cd
utest@server:~$ touch /tmp/acl_test/utest.txt
utest@server:~$ ls -ld /tmp/acl_test | logger -t 2
utest@server:~$ tail -5 /var/log/syslog
tail: cannot open '/var/log/syslog' for reading: Permission denied
utest@server:~$ sudo tail -5 /var/log/syslog
Apr 27 19:33:32 server 2: user::rw-
Apr 27 19:33:32 server 2: group::r--
Apr 27 19:33:32 server 2: other::r--
Apr 27 19:33:32 server 2:
Apr 27 19:47:20 server 2: drwxrwxrwx 2 guest guest 4096 Apr 27 19:46 /tmp/acl_test
```

```
utest@server:~$ ls -l /tmp/acl_test | logger -t 2
utest@server:~$ sudo tail -5 /var/log/syslog
Apr 27 19:33:32 server 2: other::r--
Apr 27 19:33:32 server 2:
Apr 27 19:47:20 server 2: drwxrwxrwx 2 guest guest 4096 Apr 27 19:46 /tmp/acl_test
Apr 27 19:49:10 server 2: total 0
Apr 27 19:49:10 server 2: -rw-rw-r-- 1 utest utest 0 Apr 27 19:46 utest.txt
utest@server:~$ _
```

```
utest@server:/tmp$ getfacl acl_test | logger -t 2
```

```
utest@server:/tmp$ sudo tail -10 /var/log/syslog
Apr 27 19:50:32 server 2: group::rwx
Apr 27 19:50:32 server 2: other::rwx
Apr 27 19:50:32 server 2:
Apr 27 19:50:54 server 2: # file: acl_test
Apr 27 19:50:54 server 2: # owner: guest
Apr 27 19:50:54 server 2: # group: guest
Apr 27 19:50:54 server 2: user::rwx
Apr 27 19:50:54 server 2: group::rwx
Apr 27 19:50:54 server 2: other::rwx
Apr 27 19:50:54 server 2:
utest@server:/tmp$ _
```

```
utest@server:/tmp$ getfacl acl_test/utest.txt | logger -t 2
utest@server:/tmp$ sudo tail -10 /var/log/syslog
Apr 27 19:50:54 server 2: group::rwx
Apr 27 19:50:54 server 2: other::rwx
Apr 27 19:50:54 server 2:
Apr 27 19:52:24 server 2: # file: acl_test/utest.txt
Apr 27 19:52:24 server 2: # owner: utest
Apr 27 19:52:24 server 2: # group: utest
Apr 27 19:52:24 server 2: user::rw-
Apr 27 19:52:24 server 2: group::rw-
Apr 27 19:52:24 server 2: other::r--
Apr 27 19:52:24 server 2:
```

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

touch /tmp/acl_test/prohibited.txt

Is it possible to invoke this command? - Yes

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*).

```
utest@server:~$ setfacl -m u:utest:r /tmp/acl_test/utest.txt
```

```
utest@server:~$ touch /tmp/acl_test/prohibited.txt
```

```
utest@server:~$ touch /tmp/acl_test/prohibited.txt
utest@server:~$ echo "new content" > /tmp/acl_test/utest.txt | logger -t 3
utest@server:~$ cat /tmp/acl_test/utest.txt
new content
```

```
utest@server:/tmp$ getfacl acl_test/utest.txt
# file: acl_test/utest.txt
# owner: utest
# group: utest
user::rw-
user:utest:r--
group::rw-
mask::rw-
other::r--

utest@server:/tmp$
```

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 action is allowed with **chmod** (conventional mechanism). (Hint: repeat step 3, but given the new context).

```
utest@server:~$ setfacl -m u:utest:rwx /tmp/acl_test/utest.txt | logger -t 3
utest@server:~$ sudo chmod 007 /tmp/acl_test/utest.txt | logger -t 3
```

```
utest@server:~$ touch /tmp/acl_test/prohibite.txt
utest@server:~$ echo "new content" > /tmp/acl_test/utest.txt | logger -t 3
bash: /tmp/acl_test/utest.txt: Permission denied
```

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 **setfacl** 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 **getfacl**.

```
utest@server:~$ sudo setfacl -d -m u:utest:r /tmp/acl_test/
```

```
utest@server:~$ touch /tmp/acl_test/utest2.txt
utest@server:~$ getfacl acl_test/utest2.txt
getfacl: acl_test/utest2.txt: No such file or directory
utest@server:~$ cd /tmp
utest@server:/tmp$ getfacl acl_test/utest2.txt
# file: acl_test/utest2.txt
# owner: utest
# group: utest
user::rw-
user:utest:r--
group::rwx                      #effective:rw-
mask::rw-
other::rw-

utest@server:/tmp$ _
```

```
utest@server:/tmp$ getfacl acl_test/utest2.txt | logger -t 4
utest@server:/tmp$ sudo tail -10 /var/log/syslog
Apr 27 20:17:01 server CRON[1732]: (root) CMD (   cd / && run-parts --report /etc/cron.hourly)
Apr 27 20:36:05 server 4: # file: acl_test/utest2.txt
Apr 27 20:36:05 server 4: # owner: utest
Apr 27 20:36:05 server 4: # group: utest
Apr 27 20:36:05 server 4: user::rw-
Apr 27 20:36:05 server 4: user:utest:r--
Apr 27 20:36:05 server 4: group::rwx#011#effective:rw-
Apr 27 20:36:05 server 4: mask::rw-
Apr 27 20:36:05 server 4: other::rw-
Apr 27 20:36:05 server 4:
utest@server:/tmp$
```

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 **getfacl**.

```
utest@server:/tmp$ setfacl -m m::r /tmp/acl_test/utest.txt | logger -t 6
utest@server:/tmp$ getfacl acl_test/utest.txt | logger -t 6
utest@server:/tmp$ sudo tail -10 /var/log/syslog
Apr 27 20:36:05 server 4:
Apr 27 20:38:58 server 6: # file: acl_test/utest.txt
Apr 27 20:38:58 server 6: # owner: utest
Apr 27 20:38:58 server 6: # group: utest
Apr 27 20:38:58 server 6: user::---
Apr 27 20:38:58 server 6: user:utest:rwx#011#effective:r--
Apr 27 20:38:58 server 6: group::rw-#011#effective:r--
Apr 27 20:38:58 server 6: mask::r--
Apr 27 20:38:58 server 6: other::rwx
Apr 27 20:38:58 server 6:
utest@server:/tmp$ _
```

7. Delete all ACL entries relative to the */tmp/acl_test* directory.

```
Apr 27 20:38:58 server 6:
utest@server:/tmp$ setfacl -b /tmp/acl_test/ | logger -t 7
setfacl: /tmp/acl_test/: Operation not permitted
utest@server:/tmp$ sudo setfacl -b /tmp/acl_test/ | logger -t 7
utest@server:/tmp$ getfacl acl_test/ | logger -t 7
utest@server:/tmp$ sudo tail -10 /var/log/syslog
Apr 27 20:38:58 server 6: mask::r--
Apr 27 20:38:58 server 6: other::rwx
Apr 27 20:38:58 server 6:
Apr 27 20:41:04 server 7: # file: acl_test/
Apr 27 20:41:04 server 7: # owner: guest
Apr 27 20:41:04 server 7: # group: guest
Apr 27 20:41:04 server 7: user::rwx
Apr 27 20:41:04 server 7: group::rwx
Apr 27 20:41:04 server 7: other::rwx
Apr 27 20:41:04 server 7:
utest@server:/tmp$
```