6.lab. vjezba

Kreiranje novog korisničkog računa

Cilj vjezbe je upoznati se sa osnovnim postupkom upravljanja korisnickim racunima na Linux OS,s posebnim naglaskom na kontrolu pristupa.

U Linux-u svaka datoteka ili program ima vlasnika.

Svakom korisniku je dodijeljen User ID (UID).

Svaki korisnik mora pripadati barem jednoj grupi.

Grupe imaju jedinstveni Group ID (GID).

Naredbom *id* mozemo provjeriti pripadnost grupi i identifikatore UID i GID.

Mi pripadamo administratorskoj grupi *sudo.*

Kreiramo novi korisnicki racun *alice*, pomocu naredbe *adduser.* Ovo mozemo napraviti samo ako imamo administratorske ovlasti, tj pripadamo grupi *sudo*. Dakle naredba izgleda:

sudo adduser alice

```
student@DESKTOP-7Q0BASR:/mnt/c/Users/A507/$ sudo adduser alice
[sudo] password for student:
Adding user `alice' ...Adding new group `alice' (1002) ...
Adding new user `alice' (1001) with group `alice' ...
Creating home directory `/home/alice' ...Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for alice
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]home/alice
```

Nakon toga se logiramo kao novi korisnik i sa *su - alice* saznamo identifikatore i grupe kojima pripada

```
alice@DESKTOP-7Q0BASR:~$ id
uid=1001(alice) gid=1002(alice) groups=1002(alice)
```

Zatim napravimo novi korisnicki racun imena *bob.* Za ovo se trebamo naredbom *exit* vratiti u *shell* korisnika jer *alice* nema admin ovlasti.

Standardna prava pristupa datotekama

Logiramo se kao novi korisnik *alice.* U korisnikovu home direktoriju(*/home/alice*) krairamo direktorij *srp* i u njemu datoteku *security.txt* sa proizvoljnim tekstom.

```
# navigate to home directory
cd
# create a new directory
mkdir
# create a file with text
echo "Hello world" > security.txt
# print file content
cat security.txt
```

```
alice@DESKTOP-7Q0BASR:~$ mkdir srp
alice@DESKTOP-7Q0BASR:~$ cd srp
```

```
alice@DESKTOP-7Q0BASR:~/srp$ echo "Hello world" > security.txt
```

Naredbom *ls -l* dobijemo uvid u datoteke unutar direktorija, prava, velicinu i datum

```
alice@DESKTOP-7Q0BASR:~/srp$ ls -l
total 4
-rw-rw-r-- 1 alice alice 12 Jan 10 13:17 security.txt
```

iz *-rw-rw-r--* smo saznali da korisnik moze citati i pisati u dat, grupe takoder a ostali samo citati.

Ovo smo mogli ostvariti i naredbom *getfacl.*

Naredbom *chmod* cemo oduzeti pravo pristupa vlasniku datoteke datoteci *security.txt*

NEKI PRIMJERI KORISTENJA NAREDBE CHMOD

```
# Remove (u)ser (r)ead permission
chmod u-r security.txt
# Add (u)ser (r)ead permission
chmod u+r security.txt
# Remove both (u)ser and (g)roup (w)rite permission
chmod ug-w security.txt
# Add (u)ser (w)rite and remove (g)roup (r)ead permission
chmod u+w,g-r security.txt
# Add (u)ser (r)read, (w)rite permissions and remove e(x)ecute permpission
chmod u=rw security.txt
```

`alice@DESKTOP-7Q0BASR:~/srp$ chmod u-r security.txt`

Sada ne mozemo citati sadrzaj datoteke

```
alice@DESKTOP-7Q0BASR:~/srp$ cat security.txt
cat: security.txt: Permission denied
```

```
alice@DESKTOP-7Q0BASR:~/srp$ getfacl security.txt
# file: security.txt
# owner: alice
# group: alice
user::-w-
group::rw-
other::r--
```

Sada cemo se u dopunskom terminalu logirati kao bob. bob ce moci procitati datoteku jer je on ostali

```
student@DESKTOP-7Q0BASR:/mnt/c/Users/A507$ su - bob
Password:
bob@DESKTOP-7Q0BASR:~$ cat /home/alice/srp/security.txt
Hello world
```

Oduzet cemo pravo citanja bobu

```
alice@DESKTOP-7Q0BASR:~/srp$ chmod o-r security.txt

bob@DESKTOP-7Q0BASR:~$ cat /home/alice/srp/security.txt
cat: /home/alice/srp/security.txt: Permission denied
```

Sada cemo bobu omoguciti pristup sadrzaju datoteke, ali na nacin da on ima pristup samo ako je clan grupe koja je vlasnik security.txt

```
# 1. Learn the group that owns the file using getfacl command
# 2. Add new user (bob) to that group (requires administrator privileges)
usermod -aG <owner_group> bob

# 3. Logout and login for this change to take effect
# 4. Verify the group membership of bob
```

```
id
# 5. Finally, try to read the file content
```

alice nije admin pa ne moze dodati boba u grupu

```
alice@DESKTOP-7Q0BASR:~/srp$ usermod -aG alice bob
usermod: Permission denied.
usermod: cannot lock /etc/passwd; try again later.
alice@DESKTOP-7Q0BASR:~/srp$ sudo usermod -aG alice bob
[sudo] password for alice:
alice is not in the sudoers file.  This incident will be reported.
```

```
alice@DESKTOP-7Q0BASR:~/srp$ exit
logout
student@DESKTOP-7Q0BASR:/mnt/c/Users/A507/$ sudo usermod -aG alice bob
[sudo] password for student:
```

Izlogiramo se iz alice

```
bob@DESKTOP-7Q0BASR:~$ cat /home/alice/srp/security.txt
cat: /home/alice/srp/security.txt: Permission denied
bob@DESKTOP-7Q0BASR:~$ exit
```

Moramo se izlogirati i ponovno ulogirati da bi aktivirali nova prava:

```
student@DESKTOP-7Q0BASR:/mnt/c/Users/A507$ su - bob
Password:
bob@DESKTOP-7Q0BASR:~$ cat /home/alice/srp/security.txt
Hello world
```

Logirat cemo se kao jedan od dodanih korisnika i pokusati procitati dat /etc/shadow

```
bob@DESKTOP-7Q0BASR:~$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```

```
bob@DESKTOP-7Q0BASR:~$ getfacl etc/shadow
# file: etc/shadow
# owner: root
# group: shadow
user::rw-
group::r--
other::---
```

bob je other korisnik pa nema prava

ACL (Access Control Lists) - kontrola pristupa

Uklonimo boba iz grupe alice

```
student@DESKTOP-7Q0BASR:/mnt/c/Users/A507$ sudo gpasswd -d bob alice
```

U prethodnom zadatku pristup sadržaju smo omogućili dodavanjem novog korisnika u grupu koja je vlasnik predmetne datoteke. Korištenjem ACL, ovo možemo jednostavnije riješiti tako da u ACL datoteke *security.txt* dodamo novog korisnika sa *(r)ead* ovlastima (potrebne su administratorske ovlasti).

```
# 1. Read/record current permissions defined on the file
getfacl security.txt
# 2. Add (u)ser bob to the ACL list of the file with (r)ead premission
setfacl -m u:bob:r security.txt
# 3. Check the updated permissions defined on the file
getfacl security.txt
```

`# 4. Login as bob, navigate to the file and try to read its content`

`cat security.txt`.

```
# Removing one entry from ACL
setfacl -x u:bob security.txt
# Removing the complete ACL
setfacl -b security.txt
```

Sada pomocu ACL postavljamo boba u posebnu grupu.Sustav ce se voditi pravima koje ima iz ACL liste

```
student@DESKTOP-7Q0BASR:/home/alice/srp$ sudo setfacl -m u:bob:r security.txt
student@DESKTOP-7Q0BASR:/home/alice/srp$ getfacl security.txt
# file: security.txt
# owner: alice
# group: alice
user::rw-
user:bob:r--
group::rw-
mask::rw-
other::---
```

Linux procesi i kontrola pristupa

Linux procesi su programi koji se trenutno izvršavaju u odgovarajućem adresnom prostoru. Trenutno aktivne procese možemo izlistati korištnjem naredbe *ps -ef*. Primjetite da proces ima vlasnika (*UID*) i jedinstveni identifikator procesa, *process identifier PID*.

Napravimo lab_6.py kod

```
import os

print('Real (R), effective (E) and saved (S) UIDs:')
print(os.getresuid())

with open('/home/alice/srp/security.txt', 'r') as f:
    print(f.read())
```

Prava nad skriptom:

```
student@DESKTOP-7Q0BASR:~$ getfacl lab_6.py
# file: lab_6.py
# owner: student
# group: student
user::rw-
group::r--
other::r--
```

Ne mozemo otvoriti zbog korisnik student

```
student@DESKTOP-7Q0BASR:~$ python3 lab_6.py
Real (R), effective (E) and saved (S) UIDs: (1000, 1000, 1000)
Traceback (most recent call last):
  File "lab_6.py", line 5, in <module>
    with open('/home/alice/srp/security.txt', 'r') as f:
PermissionError: [Errno 13] Permission denied: '/home/alice/srp/security.txt'
```

Ako pokusamo kao bob uspjet cemo

```
bob@DESKTOP-7Q0BASR:~$ python3 /home/student/lab_6.py
Real (R), effective (E) and saved (S) UIDs: (1002, 1002, 1002)

Hello world
```

jer bob ima prava citanja

Mehanizam efektivnog vlasnika procesa

Linux koristi mehanizam efektivnog vlasnika .Uz stvarnog vlasnika procesa(RUID) pridjeljen je i efektivni vlasnik(EUID) koji kernel koristi pri provjeri pristupa.Uglavnom je RUID=EUID osim kad je program oznacen posebnim *setuid* bitom

```
# Note that in place of "x" flag, we now have "s" flag
ls -l $(which passwd)
-rwsr-xr-x 1 root root 59640 Mar 22  2019 /usr/bin/passwd

getfacl $(which passwd)# file: usr/bin/passwd
# owner: root
# group: root
# flags: s--
user::rwx
group::r-x
other::r-x
```

Treba:

Izvršiti naredbu *passwd* (kao neprivilegirani korisnik).

```
passwd
Changing password for alice.
(current) UNIX password:
# !!! NEMOJTE UNOSITI NIKAKVU LOZINKU !!!
```

U drugom terminalu izvršiti sljedeću naredbu (koja će vam ispisati tekuće procese sa njihovim stvarnim i efektivnim vlasnicima):

```
ps -eo pid,ruid,euid,suid,cmd
```

Pronaći u ispisu liniju koja odgovara programu `passwd` i prokomentirajte `RUID`, `EUID` i `SUID` polja.

Provjerimo prava koja daje passwd naredba:

```
bob@DESKTOP-7Q0BASR:~$ ls -l $(which passwd)
-rwsr-xr-x 1 root root 59640 Mar 22  2019 /usr/bin/passwd
```

s flag je specijalni flag koji daje posebna prava onome koji poziva naredbu.

Pokrenimo proces promjene lozinke kao bob

```
bob@DESKTOP-7Q0BASR:~$ passwd
Changing password for bob.
(current) UNIX password:
```

U drugoj konzoli pokrenimo naredbu `ps`. Ona sa sljedećim parametrima ispisuje tekuće procese sa njihovim stvarnim i efektivnim vlasnicima:

```
student@DESKTOP-7Q0BASR:~$ ps -eo pid,ruid,euid,suid,cmd | grep passwd
  747  1002     0     0 passwd
  750  1000  1000  1000 grep --color=auto passwd
```

Primjetimo da je efektivni id korisnika koji zove passwd naredbu 0 (root), dok je pravi id 1002. To zapravo znači da bob privremeno, odnosno samo za vrijeme izvršavanja naredbe passwd dobije prava roota, kako bi si mogao promijeniti lozinku.