SRP Vježba 6

U ovoj vježbi bavimo se kontrolom pristupa na Linux-u. U Linux-u svaki korisnik ima jedinstveni korisnički ID (UID) i pripada nekoj grupi. Svaka grupa ima svoj ID (GID).

Uz pomoć naredbe “id” provjeravamo naš UID a uz pomoć naredbe “groups” pripadnost grupama.

Za kreiranje novog korisničkog računa koristimo:

```
sudo adduser alice
```

Prijavimo se kao novi korisnik:

```
su - alice
```

Kreirajmo novi direktorij i dokument.

```
cd /home/alice/
mkdir SRP
echo "Hello World!" > Security.txt
```

Za pregled vlasnika i dopuštenja resursa koristimo:

```
getfacl Security.txt
```

Naredbom “chmod” možemo modificirati dopuštenja nad resursom. Npr: ako želimo ukloniti pravo na čitanje datoteke.

```
chmod u-r Security.txt
```

Npr: ako želimo da korisnik može samo pisati u datoteku:

```
chmod u+w Security.txt
```

Prava možemo oduzeti i neposrendo tako da oduzmemo prava čitanja i izvršavanja nad roditeljskim direktorijem.

Korisnike dodajemo u grupe na način:

```
usermod -aG ime_grupe bob
```

bob je ime korisničkog računa kojeg smo ranije kreirali.

Datoteki kreiranoj na računu alice moguće je pristupiti s računa bob zato što “others” imaju pravo čitanja. Kada bismo uklonili “others” sa liste čitatelja, pristup bi bilo moguće implementirati na način da korisnika dodamo u grupu koja ima pristup toj datoteci.

```
getfacl Security.txt
```

Vidimo da bob nema pravo pristupa datoteci.

```
setfacl -m u:bob:r Security.txt
```

Eksplicitno dodajemo korisnika bob kao čitatelja.

Sada se prijavimo kao bob i vidimo da je promjena uspješnja:

```
cat Security.txt
"Hello World!"
```

Na sličan način vršimo uklanjanje:

```
setfacl -x u:bob Security.txt
```

Gore navedeni princip zove se *Access Control List (ACL).*

Kako bismo uklonili listu u potpunosti:

```
setfacl -b Security.txt
```

Svaki proces koji se izvršava na Linux-u ima svog vlasnika (UID) i vlastiti identifikator (PID).

Uvjerimo se da bob nema pristup sadržaju datoteke Security.txt.

```
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())
```

Pokrenimo skriptu kao alice pa kao bob.

Vidimo da identifikatori nisu isti ovisno tko pokreće proces. I s obzirom da bob nema prava čitanja dobiva grešku.

Što zapravo znači efektivni ID? Zamislimo situaciju u kojoj želimo promjeniti zaporku za svoj korisnički račun. Onaj je pohranjena u /etc/shadow kojoj samo korisnik s UID=0 ima pristup. Sada se pitamo ako svaki proces dobije ID od onoga tko ga pokrene kako je moguće da korisnik izvrši komandu za promjenu zaporke (passwd) koja mijenja taj resurs? Ovo je jedan od specijalnih slučajeva u Linux okružju. Naime, postoji specijalni “setuid” bit kojim program može biti označen od strane kernel-a koji govori sistemu kontrole pristupa da privremeno promjeni efektivni ID u onaj više privilegirani kako bi ta promjena mogla biti izvršena

Mala demonstracija: (pokrenimo ovo kao alice)

```
passwd
```

Sada se u drugom terminalu prijavimo kao bob i pokrenimo:

```
ps -eo pid,ruid,euid,suid,cmd | grep passwd
```

Vidimo da je efektivni ID različit od našeg korisničkog ID-a.