

$$10 + 3 + 6 + 0 + 4 + 3 = 26$$

1. Simona Miboučiková

a.) Setuid a Setgid pre spustiteľné súbory  
↑ platí pre užívateľa      ↑ platí pre skupinu

..prístupové práva umožnia spustiť súbor. (X-execute) X majú programy a skripty  
Po vykonaní exec sa UID procesu zmení na UID vlastníka súboru  
Ak je vlastníkom root, súbor je spustiteľný i nastavené setuid  
Po exec proces beží pod root (napr. použijeme pri zmene prihlasovacej hesla)

b.) setgid pre adresáre

všetky nové súbory v adresári majú pridelenú rovnakú skupinu  
Novovzniknuté podadresáre majú zdedený setgid bit. ako adresár

10

3. ls - ukazuje nám zoznam súborov v aktuálnom adresári

a.)  $\hookrightarrow \$X$  '\$X' echo interpretuje X ako slovo

b.)  $\hookrightarrow$  vypísala všetky súbory v adresári ~~ls~~ (echo \*  $\hookrightarrow$  ls)  
\$X dereferencuje premennú x

6

d.) úvodzovky odšpeciálnujú všetko okrem \$  
\$X = znakový súbor  $\Rightarrow$  \$X \* = echo \*  
 $\hookrightarrow$  echo echo \*  
                     $\nwarrow$  výpis súborov

c.) ~~\$X~~ ~~reference~~ na  
\$(echo \*)  
 $\nwarrow$  vykonaj príkaz ktorý je v zátvorke  
echo \$X - vykoná príkaz \$X  
výpis

1) B je premenná X referuje 'echo \*'  
2) B dereferencovaním \$X dostaneme echo echo \*  
3) Preto sa rozvinie \*, takže máme echo echo ahoj.txt dovi.txt  
4) Preto príkaz sa toto vám vykoná, čiže výstup chýba je echo ahoj.txt dovi.txt

a kde čo?

⑤

run-test

script1.py

~~script1.py~~

#! bin/bash

NAZOV=\$(echo "\$1" | sed 's/\./py\$/')

#

if test ! -d NAZOV  
then

mkdir NAZOV

~~fi~~

~~cp test-in.txt~~

cp "\$1" test-in.txt

cat \$1 > test-out.txt

fi

⑤  
 run\_test script1.py ~~script1.py~~  
 #!/bin/bash  
 NAZOV=\$(echo "\$1" | sed 's/\.py\$//')

4

if test ! -d \$NAZOV  
 then

mkdir NAZOV

~~cp test-in.txt~~

cp "\$1" test-in.txt

cat \$1 > test-out.txt

fi

jeden alebo  
 viac argumentov!

bez koncovky

môže byť

o tom  
 rozhodne!

to je správne?

⑥  
 def txt-to-words(txt):

x = txt.split()

to je veľa  
 riadkov/riadkov  
 nie jeden!

for i, item in x

if x[i] == "," or x[i] == ";"

x.remove(i)

continue

new x = x[i]

3

⑥

ask

def txt-to-words(txt):

x = txt.split()

for i, item in x

if x[i] == " " or x[i] == "."

x.remove(i)

continue

new\_x = x

- (1) (10 bodov) Vysvetlite, čo znamenajú prístupové bity
- (a) `setuid` a `setgid` pre spustiteľné súbory,
  - (b) `setgid` pre adresáre.
- (2) (10 bodov) Na základe vašej znalosti použitých systémových funkcií vysvetlite, čo a ako robí funkcia `forkmap`. Na riadkoch 35–40 vidíte príklad jej použitia. Vaša odpoveď by sa nemala skladať z viet typu „na riadku  $n$  sa volá funkcia `foo`“ (pretože taká odpoveď preukazuje iba to, že viete čítať) ale mala by preukazovať vaše porozumenie veci.
- Je to napísané v Pythone ale to by pre Vás nemal byť problém.
- Ako dlho (rádovo) si myslíte, že ten program beží?

```

1  import os      importujeme module
2  import sys
3  import time
4
5  def forkmap(f,zoz):
6
7      read_end,write_end=os.pipe() ← vytvára rúru s čítavým a zapisujúcim
8      read_file=os.fdopen(read_end) ← do premennej read_file      deskriptorom
9      write_file=os.fdopen(write_end,"w") ← odtiaľ čítavý koniec
10
11  → for i,item in enumerate(zoz): ← začína prvok zo zoznamu
12      if os.fork()==0: nastane, ak sme v child procese
13          write_file.write("%d %d\n" % (i,f(item))) ← odtiaľ súbor na zápis
14          write_file.flush() ← vymazanie      R zápis do súboru
15          sys.exit() ← ukončujú "medzi pamäť"
16
17      write_file.close() ← zatváranie súboru na zápis
18
19      for x in zoz:
20          os.wait() ← čaká na dokončenie child procesu
21
22      zoz_ret=[0]*len(zoz)
23
24      while True:
25          line=read_file.readline() ← z read_file sa číta po riadkoch
26          if not line: ← ak už nie je čo prečítať program sa ukončí
27              break
28          i_s,result_s=line.split() ← slova v riadku rozparuje
29          zoz_ret[int(i_s)]=int(result_s) ← do zoznamu sa uložia jednotlivé znaky
30
31      return zoz_ret
32
33  def square(x):
34      # Dlhý výpočet
35      time.sleep(5)
36      return x*x
37
38  print(forkmap(square,range(100)))

```

1  
2  
...

0\*0  
1\*1  
...

OK, ALĚ ČO TO CELĚ ROBI,  
AKO TO FUNGUJE?

- (3) (10 bodov) Uvažujme takúto situáciu: máme spustený bash na termináli, príkaz `ls` vypisuje toto:

1

0\*0  
1\*1  
...

ROBI,

3

- (1) (10 bodov) Vysvetlite, čo znamenajú prístupové bity
  - (a) `setuid` a `setgid` pre spustiteľné súbory,
  - (b) `setgid` pre adresáre.
- (2) (10 bodov) Na základe vašej znalosti použitých systémových funkcií vysvetlite, čo a ako robí funkcia `forkmap`. Na riadkoch 35–40 vidíte príklad jej použitia. Vaša odpoveď by sa nemala skladať z viet typu „na riadku  $n$  sa volá funkcia `foo`“ (pretože taká odpoveď preukazuje iba to, že viete čítať) ale mala by preukazovať vaše porozumenie veci.  
 Je to napísané v Pythone ale to by pre Vás nemal byť problém.  
 Ako dlho (rádovo) si myslíte, že ten program beží?

```

1  import os      importujeme module
2  import sys
3  import time
4
5  def forkmap(f,zoz):
6
7      read_end,write_end=os.pipe()
8      read_file=os.fdopen(read_end)
9      write_file=os.fdopen(write_end,"w")
10
11     for i,item in enumerate(zoz):
12         if os.fork()==0:
13             write_file.write("%d %d\n" % (i,f(item)))
14             write_file.flush()
15             sys.exit()
16
17     write_file.close()
18
19     for x in zoz:
20         os.wait()
21
22     zoz_ret=[0]*len(zoz)
23
24     while True:
25         line=read_file.readline()
26         if not line:
27             break
28         i_s,result_s=line.split()
29         zoz_ret[int(i_s)]=int(result_s)
30
31     return zoz_ret
32
33 def square(x):
34     # Dlhý výpočet
35     time.sleep(5)
36     return x*x
37
38 print(forkmap(square,range(100)))

```

pre každý prvok zo zoznamu (iteruje)

vytvára rúru s čítavým a zapisujúcim deskriptorom

do premennej `read_file` otvára čítavý koniec

otvára súbor na zápis

hovorí "medzi pamäť"

ukončí proces

otvára súbor na zápis

čaká na dokončenie child procesu

získava v riadku rozparuje

do zoznamu sa uložia jednotlivé znaky

1  
2

funkcia zapisuje do súboru čísla od 0 po 100 a vyhodnotí aplikuje na ne funkciu `square`

- (3) (10 bodov) Uvažujme takúto situáciu: máme spustený bash na termináli, príkaz `ls` vypisuje toto:

1

0\*0  
1\*1  
:  
n\*n