Programska podrška mjernih i procesnih sustava

vježba br. 7: **MREŽNI SERVISI**

ZADACI ZA VJEŽBU

1. MREŽNI SERVISI

Mrežni servisi su *daemon* programi koji se nalaze na server računalu. Oni udaljenom korisniku omogućavaju pristup dotičnom računalu. Udaljeni pristup računalu vrši se preko mrežnih portova (TCP ili UDP). Mrežni portovi su zapravo softverska apstrakcija kojom se omogućuje da se različiti paketi koji pristižu na mrežnu karticu uređaja određene IP adrese proslijeđuju različitim programima. Svaki paket koji stigne ima u sebi i polje u kojem je upisan broj koji predstavlja port za koji je paket namijenjen.

Serverski programi (servisi) osluškuju svoje mrežne portove i točno je određeno koji port pripada kojem mrežnom servisu. Npr. web server nalazi se na portu 80, dok se telnet ostvaruje preko porta 23. Popis svih korištenih servisa i portova koji im odgovaraju nalazi se u konfiguracijskoj datoteci `/etc/services`. Detaljne specifikacije o tome koji je servis po dogovoru pridružen kojem portu mogu se naći u dokumentu RFC 1340.

- Proučite datoteku `/etc/services`

Mrežni *daemon* može raditi samostalno ili kao *slave* `inetd` *daemona*. Nazivi *daemona* u pravilu završavaju slovom `d`. Tako se npr. ftp *daemon* zove `ftpd` (`in.ftpd`), a telnet *daemon* `telnetd` (`in.telnetd`). Svaki mrežni *daemon* može dakle raditi samostalno, no ukoliko na jednom računalu imamo puno servisa kojima želimo pristupati, nije mudro da svi oni rade samostalno jer se time povećava opterećenje sustava.

Zbog toga se koristi `inetd` server koji cijelo vrijeme osluškuje portove računala. Ukoliko na nekom od portova nađe uspostavljenu vezu, pokreće odgovarajući *daemon* program. `inetd` se naziva i internet "super-server". Koje će daemone `inetd` pokretati zapisano je u datoteci `/etc/inetd.conf`. Više o tome može se pročitati u dodatnim materijalima.

- Proučite datoteku `/etc/inetd.conf`. Uočiti da se uz ime servisa ne navodi i broj porta kojem odgovara. Sjetite se tko je zadužen za pridjeljivanje broja porta određenom servisu.
- Koje servise pokreće `inetd` na računalu na kojem radite i kojim portovima odgovaraju?

2. ELEKTRONIČKA POŠTA

Za slanje elektroničke pošte najčešće se koristi SMTP protokol (*Simple Mail Transfer Protocol*). Primanje se elektroničke pošte vrši POP3 protokolom (*Post Office Protocol*, verzija 3). Prema dogovoru, SMTP koristi port 25, a POP3 port 110. Ovo su tekstualni protokoli i vrlo ih je jednostavno koristiti. U ovom ćete dijelu vježbe vidjeti što uistinu radi vaš mail klijent (npr. MS Outlook) kada šaljete ili primate poruke.
Pošaljimo prvo jednu poruku telnetom na port 25 (SMTP):

```
 [17:20] ~% telnet pinus.cc.fer.hr 25
<localhost>
Trying 161.53.73.18...
Connected to pinus.cc.fer.hr.
Escape character is '^]'.
220 pinus.cc.fer.hr ESMTP Sendmail 8.9.3/8.9.3; Tue, 4 Jan 2000 18:10:54 +0100 (MET)
MAIL FROM: hrvoje.bogunovic@fer.hr
250 hrvoje.bogunovic@fer.hr... Sender ok
RCPT TO: hbogunov@zesoi.fer.hr
250 hbogunov@zesoi.fer.hr... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
From: Perica Opasni <perica.incognito@bezveze.hr>
To: Miro
Cc: Pero
Subject: sumnjiva poruka
Date: malo poslije nove godine
XYZheader: ovaj header je tu iako nikome nista ne znaci

Ovo je tijelo poruke jer je od zaglavlja odvojeno jednim praznim redom.
Nemoj misliti da se ne moze otkriti tko je uistinu ovu poruku poslao i od
kuda je poslana.
       opasni haker Perica
.
250 SAA07620 Message accepted for delivery
QUIT
221 pinus.cc.fer.hr closing connection
Connection closed by foreign host.
```

Ukoliko pogriješite, a ne radi Delete tipka, pokušajte sa Ctrl+H, to je njezin ekvivalent koji će vjerojatnije funkcionirati. Kako se može vidjeti, autorizacija korisnika nije potrebna i svatko može poslati poruku sa bilo čijeg SMTP servera i pod bilo čijim imenom. No, SMTP server pamti IP adresu pošiljatelja i svaku je ovakvu zlouporabu moguće razotkriti.

Ono što unesemo uz MAIL FROM i RCPT TO naredbe neće biti vidljivo primatelju poruke, a označava stvarnog pošiljatelja i primatelja poruke. Poslije naredbe DATA dolazi ono što šaljemo. Prvi dio predstavlja zaglavlje (*header*), a drugi dio predstavlja tijelo (*body*) poruke. Kraj zaglavlja i početak tijela poruke označava se praznim retkom. Postoje pravila kako zaglavlje treba izgledati (standardna stavke: `from`, `to`, `cc`, `subject`...) no nije ih se potrebno pridržavati. Zaglavlje se može i u potpunosti izostaviti. Tijelo poruke završava se točkom u novom redu.

Naredbom QUIT prekida se veza sa SMTP serverom. Imajte na umu da se naredbe mogu pisati i malim slovima.

Upotrijebite naredbu HELP da biste dobili spisak svih raspoloživih naredbi.

- Pošaljite sami sebi nekoliko mailova na računalo na kojem imate korisnički *account* i isprobajte različite formate zaglavlja.

3. SHELL SKRIPTE (40%)

Shell skripte su programi koji su interpretirani od strane ljuske. *Shell* skripte vrlo su često korištene za automatiziranje izvršavanja niza naredbi ljuske. Sadrže i naredbe za kontrolu toka programa kao npr. `if`, `else`, `while`...
Na UNIX sistemima često su korištene prilikom podizanja sustava i konfiguracije raznih servisa. Takve su skripte najčešće napisane u Bourne ljusci. No, skripte se mogu pisati i u bilo kojoj drugoj ljusci (`tcsh`, `csh`, `zsh`...). Od ljuske do ljuske sintaksa pojedinih naredbi (npr. kontrola toka programa) i mogućnosti variraju.
Evo primjera jedne jednostavne Bourne shell skripte koja ispisuje poruku na standardni izlaz.

```
#!/bin/sh
echo "Dobar dan!"
```

Koji će program, tj. ljuska interpretirati skriptu navedeno je u prvom retku, poslije `#!` znaka.
Da bi se skripta mogla izvršavati, potrebno joj je promijeniti parametre, dodavanjem mogućnosti izvršavanja:

```
$ chmod +x ime_skripte
```

Ukoliko je želite pokrenuti iz trenutnog radnog direktorija, učinite to navođenjem puta:

```
$ ./ime_skripte
```

Skripta se može pozvati s nekim parametrima. Ulaznim parametrima u skripti se može pristupiti preko odgovarajućih varijabli. Tako npr. varijable `$1`, `$2`, `$3` u sebi sadržavaju prvi, drugi i treći parametar. Broj parametara kojima je skripta pozvana zapisan je u `$#` varijablu. PID procesa ljuske može se doznati korištenjem varijable `$$`.

```
#!/bin/sh
echo "Dobar dan" $1 $2
echo Unijeli ste $# ulazna parametra, ali samo prva dva koristimo!
```

Evo rezultata izvršavanja gornje skripte:

```
[14:11] ~% ./skripta Ivan Pekar 123412 daasdfa
Dobar dan Ivan Pekar
Unijeli ste 4 ulazna parametra, ali samo prva dva koristimo!
```

`If` naredba ima slijedeću sintaksu:

```
if uvjet
then
     naredbe
fi
```

Ukoliko `uvjet` vrati status `true` (izlaz nula – pomalo suprotno logici iz svijeta C-a), onda se izvršava dio `naredbe`. Ukoliko je `uvjet` različit od nule, izvršava se ono poslije naredbe `fi`.

Slijedeći program provjerava da li postoji određeni korisnik u sustavu. Ime korisnika unosimo kao argument u komandnoj liniji.

```
#!/bin/sh
user=$1
if grep $user /etc/passwd
then
        echo "$user ima account"
else
        echo "$user nema account"
fi
```

Korisna naredba za testiranje uvjeta je `test` ili `[` (da, otvorena uglata zagrada je uistinu drugi način označavanja naredbe `test`). Njena sintaksa je slijedeća: `test uvjet` ili još jednostavnije `[ uvjet ]`.
Pogledati u `man` stranicama mogućnosti korištenja `test` naredbe.
Evo primjera kako se može testirati da li je varijabla `$ime` jednaka `student` te da li datoteka `/home/knoppix/program.c` postoji.

```
#!/bin/sh

if [ $1 = "student" ]
then
        echo "Unijeli ste ime student"
fi

if [ -f /home/knoppix/program.c ]
then
        echo "Datoteka program.c postoji"
else
        echo "Datoteka program.c ne postoji"
fi
```

Naredba za izvršavanje skripte u točno određeno vrijeme može imati jedan od slijedećih oblika:

```
$ at  -f ./ime_skripte now + 1 minutes
$ at  -f ./ime_skripte midnight
$ at  -f ./ime_skripte 2am tomorrow
$ at  -f ./ime_skripte now + 8 hours
```

Cilj ovoga dijela vježbe je napisati *shell* skriptu koja će, kada se pokrene, započeti skidanje datoteka sa Interneta. Pomoću naredbe `at` možemo postići da se skripta pokrene u točno određeno vrijeme, npr. u jedan sat u jutro slijedećega dana. Problem je da se ne vidi ono što tako pozvana skripta baca na `stdout`. Prije nego koristimo `at` moramo startati `atd` (daemon) sa (kao root):

```
# /etc/init.d/atd start
```

Instaliran je program `wget`. To je program za skidanje binarnih dokumenata korištenjem HTTP i FTP protokola. Vrlo je zgodan za korištenje iz shell skripte jer je neinteraktivan, tj. nije potrebna interakcija korisnika kada se jednom program pokrene (upisivanje korisničkog imena i zaporke, izdavanje naredbe get...).

- Proučite man stranice naredbe `wget`. Skinuti njome stranicu sa URL-a `ppmps.zesoi.fer.hr/index.htm`

- Napraviti *skriptu* koja će se spojit na ppmps web poslužitelj i skinuti s njega jednu datoteku.
- Skinutu datoteku snimiti pod imenom koje je korisnik dao kao prvi argument prilikom pozivanja skripte.
- Testirati da li skripta sadrži riječ koju je korisnik unio kao drugi argument prilikom poziva skripte (naredba `grep`).
- Za svaku od navedenih radnji ispisati potvrdu na standardni izlaz.
- Podesite da se skripta počne izvršavati jednu minutu nakon njenog poziva.

4. SERVER KAO SLAVE INETD SERVERA

Sada ćemo napraviti server koji će se pokretati kao slave `inetd` servera. Time je znatno pojednostavljena izrada samog serverskog programa pošto nije potrebno voditi brigu o održavanju komunikacije. Naime, sve što dolazi na odgovarajući port `inetd` preusmjerava na standardni ulaz servera, a ono što serverski program piše na svoj standardni izlaz bit će poslano na taj isti port.
Prije nego koristimo inetd moramo ga kao root pokrenuti sa:

```
# /etc/init.d/inetd start
```

Evo jednog vrlo jednostavnog primjera servera koji odgovara na dvije naredbe: help i quit. Napisan je kao skripta u Bourne ljusci.

moj_server

```
#!/bin/sh

echo 'Dobrodosli!'
brk=false
while ! $brk; do
  read command
  case $command in
    help*)
      echo 'Trenutno su podrzane samo naredbe help i quit.'
      ;;
    quit*)
      echo 'Dovidjenja'
      brk=true
      ;;
    # ovdje dodati jos naredbi
  esac
done
```

Podsjetiti se da je sintaksa `case` naredbe slijedeća:

```
case expression in
        pattern)
```

```
                commands
                ;;
        ...
esac
```

Da bi inetd znao da treba pozvati ovu skriptu kada se netko pokuša spojiti na port 5000, potrebno je u datoteku /etc/inetd.conf dodati još jednu liniju slijedećeg sadržaja:

```
5000 stream tcp nowait nobody /usr/sbin/tcpd /home/knoppix/moj_server
```

Isto tako, potrebno je skripti dodati prava izvršavanja i čitanja.

```
$ chmod 755 moj_server
```

Da bi `inetd` pročitao izmijenjenu konfiguraciju potrebno mu je poslati HUP signal:

```
# kill -HUP 129
```

gdje 129 treba zamijeniti pravim PID-om `inetd` procesa (`ps aux | grep inetd`).
Sada se možemo sa druge virtualne konzole telnetom spojiti na port 5000.

```
[11:16] ~% telnet localhost 5000
<localhost>
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Dobro dosli!
help
Trenutno su podrzane samo naredbe help i quit.
quit
Dovidjenja
Connection closed by foreign host.
```

- Prepraviti `moj_server` tako da odgovara na još dvije naredbe: `datum` i `kreni`.
- Ukoliko primi naredbu `datum`, server će klijentu poslati svoj sistemski datum (naredba `date`).
- Ukoliko primi naredbu `kreni`, odgovara sa "krećem!"