Osnovni alati za administraciju linuxa

Namjera ove knjige je da bude početno mjesto, s kojeg bi se naše novozaposlene kolege administratori CARNetovih poslužitelja na ustanovama, a i svi ostali koji to žele, mogli zaputiti u linux svijet.

Knjiga je zamišljena kao svojevrstan popis osnovnih alata kojima se linux administratori koriste u praksi, kako bi nadzirali rad poslužitelja ili detektirali i riješili eventualne poteškoće u radu. Većinom će biti govora o komandnolinijskim alatima, a uz objašnjenje rada i ispisa pojedinog alata navest ćemo i osnovne primjere korištenja.

Pozivamo sve kolege administratore da svojim prilogom, bilo člankom ili komentarom, prošire i poboljšaju ovu knjigu.

- Logirajte [1] se za dodavanje komentara

Logrotate - zaboravljeni junak

 Da su logovi, obično smješteni u direktoriju /var/log, izuzetno važni jasno je svakom sistem-inženjeru. Nema drugog načina rješavanja problema (osim ako ste problem susreli prije pa rješenje znate napamet) do povećanja razine logiranja i promatranja što smeta određenom servisu da funkcionira na željeni ili očekivani način.

Logovi, dakako, s vremenom rastu, pa ih treba brisati sa sustava (bilo bi, naravno, bolje snimati na nekakve neizbrisive medije na neko dulje vrijeme). Ovu funkciju nenametljivo obavlja program **logrotate**. On prvo zaustavi određeni servis (**prerotate**), obavi sve operacije iz konfiguracije, te ponovno starta servis (**postrotate**).

Logrotate nije jedini program te vrste, postoji primjerice *rotatelogs*, dio apache paketa.

Gotovo svaki paket brine o svojim logovima putem **conf.d** mehanizma, tako da način rotiranja i arhiviranja svojih logova stave u direktorij /etc/logrotate.d. Uzmimo za primjer /etc/logrotate.d/apt:

```
/var/log/apt/term.log {
  rotate 6
  monthly
  compress
  missingok
  notifempty
}
```

Dakle, ovdje se definira ponašanje za datoteku /var/log/apt/term.log. Što se ovdje definira? Objasnimo opcije:

rotate - broj iza opcije označava koliko će se arhiviranih datoteka čuvati. Dakle, osim trenutnog loga, čuva se još 6 starih inačica istih datoteka, term.log.1, term.log.2 itd.

monthly - vrlo očito, logovi se rotiraju jednom mjesečno, obično prvog u mjesecu. Umjesto ovog, ovdje može stajati **daily** i **weekly**.

compress - stari logovi se sažimaju programom **gzip(1)**. Ukoliko želite, možete rabiti drugi program za sažimanje. S anredbom **compresscmd** možete odrediti drugi program za sažimanje (zip, bzip2, itd). Naredba **uncompress** poništva sažimanje log datoteka, što ne vjerujemo da ćete rabiti kad postoje naredbe zless, zcat i slične.

missingok - logrotate se neće buniti ako konkretni log ne postoji. Obrnuta opcija je **nomissingok**, što je ujedno i *default* u programu logrotate.

notifempty - ukoliko je log prazan (veličine 0 bajta), neće se rotirati. Obrnuta opcija je **ifempty**, što će zbog zaglavlja kodeka za sažimanje kreirati niz malih datoteka *većih* od nula bajtova (iako je originalni log veličine 0!).

Ovo je bio vrlo jednostavan primjer, a složeniji je, primjerice, apache2:

```
/var/log/apache2/*.log {
      weekly
      missingok
      rotate 52
      compress
      delaycompress
      notifempty
      create 640 root adm
      sharedscripts
      postrotate
         if [ -f "`. /etc/apache2/envvars ; echo ${APACHE_PID_FILE:-/var/run/apache2.
pid}`" ]; then
                  /etc/init.d/apache2 reload > /dev/null
         fi
      endscript
}
```

Većinu opcija smo spomenuli, objasnit ćemo dosad nespomenute:

delaycompress - opcija određuje da se stari log ne sažima odmah, nego tek u sljedećem ciklusu, znači neće odmah postati access.log.1.gz, nego će prvo postati access.log.1, a tek onda acess.log.2.gz.

create - ukoliko želite iz sigurosnih razloga promijeniti vlasnika sažete log datoteke, onda se možete poslužiti opcijom create, a sintaksa je "*create mod vlasnik grupa*". Inače, arhivirana/rotirana datoteka ima iste atribute kao i original. Opcija se u potpunosti može anulirati opcijom **nocreate**.

sharedscripts - u slučaju da imate *wildcard* unos, baš kao u ovom slučaju, ova opcija određuje da se skripte *prerotate* i *postrotate* izvrše samo jednom. Opcija poništava opciju **nosharedscripts**, te automatski podrazumijeva uporabu opcije **create**.

postrotate - opciju smo maloprije spomenuli. Ona određuje ponašanje nakon provedbe rotiranja logova. Za apache2 ona je malo složenija, ali inače obično ovaj, ili sličan oblik:

```
/usr/bin/killall -HUP servis
```

endscript - samo označava kraj *prerotate* ili *postrotate* skripte.

Navest ćemo i treći primjer, recimo da imate neki svoj program koji ima log datoteku /var/log/program.log.

Kreirajte datoteku /etc/logrotate.d/program ovog sadržaja:

```
/var/log/program.log {
    compress
    delaycompress
    missingok
    notifempty
    rotate 4
    weekly
    dateext
}
```

Od sad će se log datoteka vašeg programa rotirati 4 puta (jednom tjedno), nakon čega će biti obrisana sa diska. Opcija **dateext** će rotiranim arhivama dati nastavak u obliku YYYYMMDD, umjesto jedostavnih brojeva .1, .2 i dalje. Ovo je jako praktično ako pravite arhive svojih logova na neke druge medije (što preporučamo!).

Kao i uvijek, za popis svih ostalih opcija konzultirajte manual stranicu sa "**man logrotate**".

- Logirajte [1] se za dodavanje komentara

pet, 2009-09-25 20:50 - Željko Borš**Kuharice:** Linux [2]
Kategorije: Operacijski sustavi [3]
Servisi [4]
Vote: 0

No votes yet

Nadzor procesa i memorije

Par riječi o procesima

Proces u linuxu predstavlja objekt preko kojeg program koristi memoriju, procesorsko vrijeme i ulazno-izlazne resurse. U kernelu postoji tablica procesa - podatkovna struktura koja bilježi razne podatke o svakom pojedinačnom procesu (status procesa, adresni prostor, roditelj procesa, otvorene datoteke, ...). Ovdje ćemo navesti samo neke od parametara procesa koji imaju važnost za uobičajeno administriranje sustava.

- PID – Identifikator procesa (process ID) – Dinamički dodijeljen od kernela, po redu, kako je proces kreiran.
- PPID – Roditelj procesa (parent PID) – Svaki proces ima PPID veći od 1, osim u slučaju inita, "majke svih procesa" koji se pokreće prilikom inicijalizacije sustava i čiji je PPID 0.

- UID – Vlasnik procesa (user ID) – Brojčani identifikator vlasnika procesa tekstualno povezan s korisnikom preko datoteke /etc/passwd (broj u trećem stupcu).
- TTY – Upravljački terminal za koji je proces vezan.

Samo postojanje procesa ne mora značiti da isti automatski zauzima procesorsko vrijeme i ostale resurse. U osnovi moguća su četiri stanja procesa:

- Aktivni proces (runnable) – proces koji se izvršava
- Spavajući proces (sleeping) – proces koji očekuje specifični događaj. Primjerice daemoni koji većinu svog vremena provedu u ovom stanju očekujući mrežnu konekciju.
- Zombi procesi (zombie) – Procesi koji pokušavaju "umrijeti" – proces je završio ali je ostao zapis u tablici procesa.
- Zaustavljeni (stopped) – Proces koji je administrativno zaustavljen.

Nadzor procesa (ps)

Osnovni alat kojim se ispisuju procesi je ps. Pomoću te naredbe možemo dobiti sve podatke o trenutnim procesima ispisane u različitim formatima. Naredba ps postoji na svim UNIX-oidima, ali se njena implementacija znatno razlikuje od sustava do sustava. Na linuxu postoji objedinjena verzija koja se ponaša različito u ovisnosti o parametrima. Generalno možemo razlikovati standardnu i BSD sintaksu.

```
sh-3.1$ ps ajxf
 PPID   PID  PGID   SID TTY      TPGID STAT   UID   TIME COMMAND
15222 15230 15222 15222 ?           -1 S     1000   0:00 sshd: ljubomir@pts/1
15230 15241 15241 15241 pts/1    15282 Ss    1000   0:00  \_ -bash
15241 15273 15273 15241 pts/1    15282 S     1000   0:00      \_ ksh
15273 15277 15277 15241 pts/1    15282 S     1000   0:00          \_ zsh
15277 15281 15281 15241 pts/1    15282 R     1000   0:00              \_ sh
15281 15282 15282 15241 pts/1    15282 R+    1000   0:00                  \_ ps ajxf
```

1.1 *Ispis ps naredbe u BSD stilu s parametrima za prikaz stabla procesa. Korisnik je više puta pokrenuo ljusku unutar ljuske i na kraju izvršio naredbu ps ajxf. Obratite pažnju na PID i PPID identifikatore.*

```
sh-3.1$ ps -ejH
  PID  PGID   SID TTY          TIME CMD
15230 15222 15222 ?        00:00:00 sshd
15241 15241 15241 pts/1    00:00:00   bash
15310 15310 15241 pts/1    00:00:00     ksh
15323 15323 15241 pts/1    00:00:00       zsh
15326 15326 15241 pts/1    00:00:00         sh
15327 15327 15241 pts/1    00:00:00           ps
```

1.2 *Ista naredba pokrenuta u standardnom linux stilu (Zanimljivo je primjetiti razliku pri korištenju crtice ispred parametara kod ovog tipa sintakse).*

Kod uobičajenog linux kernela, bilo koji korisnik može dobiti izlist svih procesa na računalu. CARNet-

ova standardna distribucija bazira se na Grsecurity dodacima kernelu kojima je običnom korisniku dozvoljen samo ispis vlastitih procesa, a za ispis svih procesa potrebno je koristiti root ovlasti.

Jedan od načina provjere procesa po potrošnji memorije je korištenje standardne naredbe ps aux uz dodatak parametara za sortiranje po količini memorije

```
korisnik@stroj:~$ sudo ps aux --sort -rss
USER          PID       %CPU     %MEM       VSZ          RSS
clamav        3205      0.1      11.9     133256      123480
amavis       17707      0.0      7.4       89084       77528
```

Stupac RSS (Resident Set Size) predstavlja dio memorije pojedinog procesa u RAM-u, a VSZ (Virtual memory SiZe) je veličina virtualne memorije procesa – obuhvaća RSS, memoriju u swapu te na file sistemu. Izražene vrijednosti su u kilobajtima.

- Logirajte [1] se za dodavanje komentara

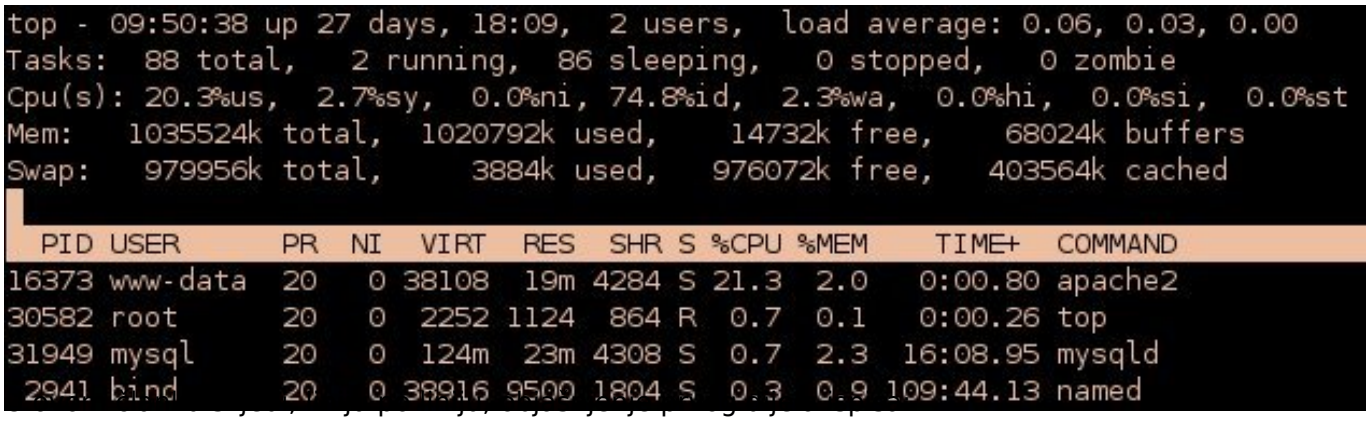
pon, 2009-05-04 13:30 - Ljubomir Hrboka**Vote:** 4

Vaša ocjena: Nema Average: 4 (1 vote)

Naredba top (1)

Ukoliko želite provjeriti kako se u realnom vremenu troše resursi na vašem poslužitelju, praćenje procesa i memorije olakšat će vam naredba top. U sljedećih par članaka pregledat ćemo mogućnosti i ispis naredbe te objasniti njene pojedine djelove.

```
sudo top
```



1. linija

top - 09:50:38 up 27 days, 18:09, 2 users, load average: 0.06, 0.03, 0.00

09:50:38 – trenutno vrijeme
up 27 days, 18:09 – poslužitelj je u pogonu već 27 dana 18 sati i 9 minuta
2 users – trenutni broj korisnika
load average: 0.06, 0.03, 0.00 – prosječan broj aktivnih procesa u zadnjih 1, 5 i 15 minuta

Ova linija je praktički ispis linux naredbe uptime. Prikaz ove linije u naredbi top može se isključiti/uključiti pritiskom na tipku l. Zanimljivo je obratiti pozornost na parametar "load average"koji nam govori koliko je procesa u navedena tri intervala koristilo ili tražilo korištenje CPU-a. Taj broj je prilično relevantan pokazatelj opterećenja poslužitelja. Kako bi analizirali njegovu vrijednost, broj je potrebno normalizirati s brojem procesora.

Primjerice, load average vrijednosti 1 na jednoprocesorskom računalu je idealno opterećenje procesora. To bi značilo da je procesor u potpunosti opterećen, ali i da su svi procesi koji su zatražili njegovo korištenje to vrijeme i dobili. Ukoliko je broj veći od jedan, to znači da je broj procesa koji su zatražili korištenje procesora veći nego što mu dopuštaju vremenski intervali korištenja, odnosno procesor je zagušen zahtjevima. U našem slučaju je average load vrlo mali, gotovo blizu nule, što bi značilo da je ovaj poslužitelj u trenutku nastanka screenshota prilično neopterećen.

Na višeprocesorskom računalu možemo zamisliti sljedeću situaciju: load average: 1.39, 2.62, 2.45. Ukoliko se, primjerice, radi o četveroprocesorskom računalu možemo kazati da je prosječna potražnja za procesorskim vremenom u zadnjih 1 minutu bila otprilike 35%, u zadnjih 5 minuta otprilike 65%, a u zadnjih 15 minuta otprilike 61%, što bi značilo da su procesori prosječno opterećeni.

2. linija

Tasks: 88 total, 2 running, 86 sleeping, 0 stopped, 0 zombie

Prikazuje broj aktivnih, spavajućih, zombie i zaustavljenih procesa.

3. linija

Cpu(s): 20.3%us, 2.7%sy, 0.0%ni, 74.8%id, 2.3%wa, 0.0%hi, 0.0%si, 0.0%st

Prikazuje postotak vremena koje je procesor proveo izvršavajući aktivne procese. Ovo je razbijeno na postotke za pojedina područja:
user mod (us) – nesistemski (korisnički) programi – Ovdje bi trebao biti najveći dio vremena

procesora. Tu su uključeni svi korisnički programi poput web, mail ili ftp servisa. Uobičajeno je da se ovaj parametar dinamički mijenja ovisno o količini korisnika na poslužitelju. Primjerice, ukoliko dosta posjetitelja istovremeno posjeti web stranice koje servisira naš poslužitelj, ovaj parametar će, u tom trenutku, znatno porasti.
sistemski modu (sy) – sistemski programi (kernel) – Ovaj postotak bi se u normalno radu trebao zadržati na malim vrijednostima, pa njegovo povećanje može biti indikacija problema.
nice (ni) – vrijeme provedeno na programe čiji je prioritet izvršenja promijenjen naredbom nice
idle (id) – neiskorišteno procesorsko vrijeme – Veliki idle time ne znači ujedno i da je poslužitelj podopterećen. Ukoliko se na poslužitelju slabijih procesorskih mogućnosti dogodi prethodno navedena situacija kada, primjerice, 20-tak korisnika u istoj sekundi zatraže dinamički generiranu web stranicu, vrlo je velika vjerovatnost da će idle time u tom trenutku pasti na 0, a poslužitelj neće biti u stanju pravovremeno pružiti uslugu.
CPU IO vrijeme čekanja (wa) – postotak vremena koje procesor provede u čekajući na IO događaje (npr. čitanje s diska ili mreže). Velike vrijednosti mogu ukazivati da eventualni problem u radu ima veze s, primjerice, diskom.
hardwareski interrupti (hi) – vrijeme provedeno na posluživanje hardwareskih interrupta
softwareski interrupti (si) - vrijeme provedeno na posluživanje softwareskih interrupta
steal time (st) – vrijeme virtualnog procesora provedeno u nenamjernom čekanju dok hypervisor poslužuje drugi virtualni procesor

Druga i treća linija mogu se u radu isljućiti/uključiti pritiskom na tipku t.

4. i 5. linija

Ove linije su ekvivalent naredbi free. Četvrta linija prikazuje stanje memorije poslužitelja. U našem slučaju vidi se da poslužitelj ima 1GB ukupne memorije, Sljedeći stupac pokazuje da je zauzeto 1020792k, a treći da je 14732k slobodno. Ovdje treba napomenuti da je zazuzeće memorije kompleksan podatak. U zauzetu memoriju linuxa treba ubrojiti, bolje rečeno, oduzeti posljednji stupac iz prikaza. Naime, stanje zauzete memorije ubraja i buffers prikazan u zadnjem stupcu četvrte linije te cached prikazanu u zadnjem stupcu pete linije. Linux dinamički alocira buffer i cached memoriju po potrebi, primjerice pri svakom čitanju podataka s diska, a s nakanom da se što više podataka drži u cache-u kako bi im se omogućio brži pristup. U slučaju da neka korisnička aplikacija zatraži memoriju, linux smanjuje cache memoriju i dodjeljuje je aplikaciji. Dakle, podatak da naše računalo ima samo 14732k slobodno u stvari nije točan, jer se bilo kojem programu u svakom trenutku može osloboditi dio memorije u cacheu.

Možda čak bitniji podatak za kvalitetan rad računala je swap iz linije 5. Naime, računalo s dovoljnom količinom memorije ne bi nikad trebalo puniti swap memoriju na disku. Povećanje used swap memorije govori o tome da aplikacije zahtjevaju više memorije od postojeće, te da bi administrator trebao razmisliti o nadogradnji RAM-a.

Četvrta i peta linija mogu se u radu isljućiti/uključiti pritiskom na tipku m.

U sljedećem nastavku ćemo analizirati ispise pojedinih procesa, te način na koji taj ispis prilagoditi potrebama.

- Logirajte [1] se za dodavanje komentara

čet, 2009-05-28 10:27 - Ljubomir Hrboka**Vote:** 5

Vaša ocjena: Nema Average: 5 (1 vote)

Naredba find


Naredba **find** je, ako nijedna druga naredba nije, "švicarski nožić" za svakog sistem-inženjera. Svi su je rabili, neki samo za osnovne operacije, a neki i za daleko složenije situacije. Bilo kako bilo, pravo je čudo da joj nismo posvetili više prostora na Portalu za sistemce.

Naredba find ima toliko opcija, da zaista nema smisla opisivati svaku od njih. Zapravo, ima, ali ćemo to napraviti preko primjera. Za detalje konzultirajte sami man ili info stranice, te wiki stranice i blogove. Ukoliko budemo opisivali baš svaku opciju finda, mogli smo jednostavno prevesti cijelu man stranicu.

Krenut ćemo od jednostavnijih primjera (naravno da naredbe ne moraju biti izvršavane kao korisnik root, ali u suprotnom dobivat ćete poruke o greškama "*cannot open file*" i slično):

```
# find /home -user marko
```

- pretražuju se sve datoteke korisnika "marko", i to samo u /home direktoriju. Ukoliko niste korisnik root, dodajte "2>/dev/null" na kraju naredbe, kako ne biste gledali poruke o nemogućnosti otvaranja određenih direktorija ili datoteka.

```
# find /var/www -name '*.pdf' -printf "%s %p\n"
```

- pretražuju se sve PDF datoteke u direktoriju /var/www i ispisuju njihove veličine u bajtovima

```
# find /var/www -name '*.pdf' -exec ls -hs '{}' \;
```

- slično kao i prethodni primjer, samo u starijoj sintaksi koja će raditi i na drugim Unixima. Možete rabiti i "linuksaški" način, koji daje opet malo drugačiji ispis (koji je ujedno i *default*, dakle opcija -print je uvijek aktivna):

```
# find /var/www -name '*.pdf' -print
```

Opcija **-printf** je preuzeta direktno iz jezika C, pa imate mnoge mogućnosti ispisa podataka o datotekama. Tako primjerice možete ispisati tip datotečnog sustava na kojem se datoteka nalazi (%F), vrijeme zadnjeg pristupa (%a), vrijeme zadnje promjene datoteke (%t), tip datoteke (%y, %Y) i tako dalje. Općenito, ispis vremena je vrlo konfigurabilan, a možete ispisati i mnoge podatke koje naredba **ls** standardno ne ispisuje.

Uzmimo par primjera iz CARNetovih paketa:

```
# find /var/lib/amavis/virusmails -mtime +7 -name "spam-*" -delete
```

- pretražuje sve datoteke starije od 7 dana (modificirane prije najmanje 7 dana) i imena koje počinje

sa 'spam-' u navedenom direktoriju i briše ih. Ime se mora zaštititi navodnicima kako ljuska (*shell*) ne bi vršila ekpanziju specijalnih znakova. Umjesto opcije -mtime, možete upotrijebiti -ctime (zadnja promjena statusa, odnosno *inodea*) i -atime (zadnje vrijeme pristupa).

Za zadnjom opcijom budite oprezni, jer je na mnogim datotečnim sustavima osvježavanje *access time* atributa (*noatime*) onemogućen jer usporava rad, i nema uvijek velike koristi.

Ukoliko smo htjeli naći datoteke novije od 7 dana, upotrijebili bismo "-7", a za stare točno 7 dana stavili bismo samo "7". Slična logika se primjenjuje i sa veličinom datoteka.

```
# find /var/lib/paket -type d -exec chown korisnik:korisnik '{}' \; -exec chmod 2775
'{}' \;
```

- pretražuje sve direktorije (opcija *d*) i mijenja vlasništvo u vlasništvo korisnika "korisnik" i grupu korisnik. Time održavatelj paketa vjerojatno popravlja "grijehe" iz prošlosti, kao i osigurava da će njegov paket raditi, čak i ukoliko sistem-inženjer nešto u međuvremenu promijeni. Ovo, inače, nije uobičajena praksa kod paketa, paketi ne bi smjeli *upadati* jedni drugima u područje. Opcija type prima i druge parametre: "-f" ("obična" datoteka), "-l" (simbolički link), "-p" (socket FIFO - *named pipe*) itd.

Naredba find se, možda, najčešće rabi kad treba pronaći datoteku određenog korisnika, grupe korisnika, datoteku određenih atributa ili tipa, te ih eventualno promijeniti, obrisati ili slično.

Nadalje, pretraživanje možemo ograničiti na razne načine:

```
# find /etc -maxdepth 1 -type f -name *.conf"
```

- pretražuje sve datoteke s nastavkom .conf, ali samo dubine do jednog direktorija, dakle bit će pronađene samo datoteke unutar /etc, ali ne i unutar /etc/apache2.

```
# find /etc -name pojam -prune
```

- pretražuje samo datoteke, a ukoliko ime odgovara nekom direktoriju, find neće "sići" dolje u taj direktorij. Ova opcija nije kompatibilna s opcijom **-depth**, pa se ova opcija ignorira kad se istodobno rabe opcije **-depth** ili **-delete**. Opcija -depth označava da se prvo obrađuju podaci u direktoriju, a tek onda sam direktorij. Opcija -delete automatski uključuje opciju -depth.

```
# find /home \( -name "*.mp3" -o -name ".vfw" \) -print
```

- pretražuje sve korisničke direktorije u potrazi za datotekama s nastavcima *.mp3 ili *.vfw. Moguće su i složenije konstrukcije:

```
# find /home \( -name "*.jpg" -a \! -size +1 \) -print
```

- ovo će ispisati sve datoteke s nastavkom *.jpg i nisu veće od 1 kilobajta.

Umjesto opcije -print , možete rabiti -ls, koja će sve "čudne" (radi se o našim znakovima "šććđž") znakove ispisati na ovaj način (u stilu jezika C):

```
# find . -ls
./\271\350\346\360\276
# find . -print
```

```
./????
```

No, to nije sve, postoji i dosta često rabljena opcija -print0, koja umjesto znaka za kraj retka (\n) ispisuje NULL znak (ASCII 000), što efektivno izgleda kao da je sve ispisano u jednom redu, umjesto u više njih. Ovo je idealno za naredbe koje moraju svoje parametre primiti u jednom retku.

Opcija -print0 vrlo često ide uz još jednu zanimljivu naredbu (**xargs**), jer omogućava da se riješi problem poruke "*Argument list too long*", ukolko u direktoriju /putanja postoje na tisuće datoteka. Primjerice, naredba:

```
# rm `find /putanja -type f`
```

može prouzročiti navedenu poruku, dok oblik:

```
# find /putanja -type f -print0 | xargs -0 rm
```

neće imati taj problem, jer listu argumenata razdvaja na manje dijelove i izbjegava probleme s ograničenjima ljuske i drugih limita. Više o naredbi xargs pročitajte na stranici http://sistemac.carnet.hr/node/826 [5].

Voljeli bismo da je to sve što smo imali reći o naredbi find, ali nije. Nažalost, ovom naredbom bismo se mogli baviti kroz cijelu seriju članaka, ali najbolje će biti da iz vlastitih primjera i iskustva naučite još više o findu. U komentaru na članak slobodno pitajte za neku konstrukciju koja vam nije jasna, pa ćemo pokušati dati odgovor.

Isto tako, slobodno dopišite svoje vlastite zgodne primjere.

- Logirajte [1] se za dodavanje komentara

uto, 2009-09-29 19:57 - Željko Borš**Kuharice:** Linux [2]
Kategorije: Software [6]
Vote: 0

No votes yet

Telnetiranje na web poslužitelj

Web poslužitelj isporučuje klijentu stranice koristeći HTTP protokol. To je protokol aplikacijskog nivoa, specificiran dokumentom RFC2616 [7]. Iako je u navedenom dokumentu specificiran čitav niz metoda koje protokol nudi, u ovom članku navest ćemo kao primjer samo dvije - GET i HEAD.

Uobičajena uspostava veze na web poslužitelj ide tako da se klijent spoji na port usluge 80 i web poslužitelju uputi zahtjev za traženom stranicom. Ukoliko tražena stranica postoji, te ukoliko nema nekih drugih prepreka za isporuku, web poslužitelj šalje klijentu traženi dokument (ukoliko je bila riječ o html stranici) ili generiranu stranicu (ukoliko je u pitanju nekakav server-side jezik). Klijent prima niz znakova - HTML kod – te ih korisniku prikazuje na monitoru u obliku tražene stranice.

Korištenjem telneta moguće je kompletno obaviti opisanu proceduru. Jedini nedostatak je nemogućnost vizualnog uobličavanja stranice od strane telnet programa. Dakle, kao rezultat radnje dobije se niz znakova koji čine HTML kod.

Da previše ne teoretiziramo, pogledajmo primjer. Sve što nam je potrebno je otvoriti Command Prompt ili se ssh klijentom spojiti na linux poslužitelj. U komandnoj liniji upišemo naredbu:

```
telnet www.example.com 80<CR>
```

Napomene:
- www.example.com je domena rezervirana upravo za primjere (RFC2606 [8]). U praksi ju je potrebno zamijeniti sa stvarnim poslužiteljem.
- <CR> označava tipku Enter.

Ukoliko smo ovo uradili iz linux terminala dobijem sljedeći odziv:

```
stroj:/# telnet www.example.com 80<CR>
  Trying 127.0.0.1...
  Connected to www.example.com.
  Escape character is '^]'.
```

Ako smo isto uradili iz Windows Command Prompta, u odzivu nam se pojavio samo trepereći kursor. Sada bi bio dobar trenutak za nekoliko kratkih uputa.

Prvo, bez obzira na to koji OS koristimo, i u jednom i u drugom slučaju kucanjem naredbe telnet pokrećemo telnet ljusku. Da bi komunicirali s telnet ljuskom potrebno je poslati tzv. escape karakter. Na računalima s hrvatskim postavkama taj karakter uobičajeno dobijemo istovremenim pritiskom tipki CTRL i č.

Drugo, kod windows telneta nije postavljen lokalni eho pa nam se prilikom tipkanja karakteri ne ispisuju na ekranu. Potrebno je ukucati CTRL+č, te nakon toga, u telnet ljusci upisati

```
set localecho<CR>
<CR>
```

Drugi Enter - u praznoj liniji - vraća nas u telnet konekciju.

Sada smo spremni za slanje naredbi web poslužitelju. Za početak je najjednostavnije uputiti običan zahtjev HTTP 1.0 protokolom - starijom i jednostavnijom verzijom protokola.

```
GET / HTTP/1.0
```

Kratak opis:
GET – komanda HTTP protokola kojom zahtjevamo dokument
/ - lokacija dokumenta - u ovom slučaju root web poslužitelja (index.html, index.php, ...)
HTTP/1.0 – ime i verzija protokola

Ispis komunikacije slijedi:

```
stroj:~$ telnet www.example.com 80
  Trying 127.0.0.1...
  Connected to www.example.com
  Escape character is '^]'.
```

```
  GET / HTTP/1.0

  HTTP/1.1 200 OK
  Date: Fri, 16 Mar 2007 13:43:30 GMT
  Server: Apache/1.3.33 (Debian GNU/Linux) PHP/4.3.10-19 mod_ssl/2.8.22 OpenSSL/0.9.7
e
  Last-Modified: Fri, 02 Mar 2007 09:50:07 GMT
  Accept-Ranges: bytes
  Content-Length: 96
  Connection: close
  Content-Type: text/html

<html>
<head>
...
...
...
</html>

  Connection closed by foreign host.
  stroj:~$
```

Minimalna komunikacija naprednijom verzijom protokola zahtjeva dodatno tipkanje nekolicine parametara:

```
stroj:~$ telnet www.example.com 80
  Trying 127.0.0.1...
  Connected to www.example.com
  Escape character is '^]'.
  GET /index.html HTTP/1.1
  Host:www.example.com
  Connection:Close

  HTTP/1.1 200 OK
  Date: Fri, 16 Mar 2007 14:30:56 GMT
  Server: Apache/1.3.33 (Debian GNU/Linux) PHP/4.3.10-19 mod_ssl/2.8.22 OpenSSL/0.9.7
e
  Last-Modified: Fri, 02 Mar 2007 09:50:07 GMT
  Accept-Ranges: bytes
  Content-Length: 96
  Connection: close
  Content-Type: text/html

<html>
<head>
...
...
...
</html>

  Connection closed by foreign host.
  stroj:~$
```

Postoji čitav niz dodatnih parametara definiranih dokumentom rfc2616, od kojih ćemo navesti samo nekolicinu važnijih:

Accept – tip podataka koje ste spremni prihvatiti
Accept-Charset – karakter set koji ste spremni prihvatiti

Accept-Language – jezici koje ste spremni prihvatiti
User-Agent – informacija o vašem pregledniku

Npr, promjenom polja User-Agent može se lažno predstaviti telnet program kao neki drugi preglednik:

User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

Ukoliko želimo zatražiti od poslužitelja samo isporuku zaglavlja, to možemo uraditi naredbom HEAD:

```
stroj:~$ telnet www.example.com 80
  Trying 127.0.0.1...
  Connected to www.example.com
  Escape character is '^]'.
  HEAD / HTTP/1.1
  Host: www.example.com
  Connection: close

  HTTP/1.1 200 OK
  Date: Fri, 16 Mar 2007 14:35:00 GMT
  Server: Apache/1.3.33 (Debian GNU/Linux) PHP/4.3.10-19 mod_ssl/2.8.22 OpenSSL/0.9.7
e
  Last-Modified: Fri, 02 Mar 2007 09:50:07 GMT
  Accept-Ranges: bytes
  Content-Length: 96
  Connection: close
  Content-Type: text/html

  Connection closed by foreign host.
  stroj:~$
```

Toliko o telnetiranju na port 80. Na kraju možemo samo konstatirati kako telnet klijent i nije baš neki dobar preglednik, ali je i više nego dovoljan za testiranje konekcije i odziva web poslužitelja.

- Logirajte [1] se za dodavanje komentara

čet, 2007-03-22 15:23 - Ljubomir Hrboka**Kuharice:** Za sistemce [9]
Kategorije: Servisi [4]
Vote: 0

No votes yet

Uvod u telnet

Telnet je komunikacijski protokol čija je primarna uloga osigurati standardnu metodu povezivanja terminala i terminalno orijentiranih procesa. U svakodnevnoj uporabi izraz telnet povezujemo s istoimenim programom koji koristi telnet protokol. To je vjerojatno najstariji internet program koji je još danas u uporabi – preteča telneta potječe iz 1969. godine.

Kada kažemo - telnetirati se negdje – najčešće mislimo na uspostavu veze između našeg računala i udaljenog poslužitelja, kako bismo na udaljenom poslužitelju obavili nekakve radnje, poput čitanja elektroničke pošte ili promjene lozinke. Pri tome se na udaljeni poslužitelj uobičajeno spajamo na port usluge 23. Takva se tip komunikacije danas, smatra zastarjelim, štoviše, iz sigurnosnih razloga nepoželjnim, jer se svi podaci, uključujući i lozinke, prenose kao čisti tekst.

Međutim, korištenjem telneta moguća je uspostava konekcije na bilo koji port usluge na udaljenom ili lokalnom računalu, te, ukoliko usluga to podržava, razmjenjivanje kontrolnih poruka s poslužiteljem. Poznavanjem kontrolnih poruka pojedinog protokola može se izvršiti kompletna konekcija na pojedinu uslugu, od samog dogovaranja pristupa, do korištenja iste.

Drugim riječima, ukoliko želite na brzinu provjeriti da li vam radi web, mail, ftp ili neki drugi servis, vjerojatno najbrži način je upravo korištenjem telneta. To je dodatno olakšano činjenicom što je telnet program, barem do sada, bio dostupan u osnovnoj instalaciji svakog operacijskog sustava. Pišući ove retke, malo sam se iznenadio činjenicom da je, u novim verzijama Microsoftove Viste, izbačen iz osnovne instalacije, odnosno da ga je, pri instalaciji, potrebno zasebno dodati. Naravno, osim komandnolinijskog telneta, postoji i niz telnet programa s više mogućnosti među kojima je, od besplatnih, najpoznatiji putty (telnet/SSH klijent) [10].

U sljedećih nekoliko članaka pokušat ću, za svaki protokol pojedinačno, sistematizirati troubleshooting telnetom najčešće korištenih protokola (HTTP, FTP, SMTP, POP, IMAP, ...).

- Logirajte [1] se za dodavanje komentara

pet, 2007-03-16 13:19 - Ljubomir Hrboka**Kuharice:** Za sistemce [9]
Kategorije: Servisi [4]
Vote: 0

No votes yet

Source URL: https://sysportal.carnet.hr/node/571

Links
[1] https://sysportal.carnet.hr/sysportallogin
[2] https://sysportal.carnet.hr/taxonomy/term/17
[3] https://sysportal.carnet.hr/taxonomy/term/26
[4] https://sysportal.carnet.hr/taxonomy/term/28
[5] https://sysportal.carnet.hr/node/826
[6] https://sysportal.carnet.hr/taxonomy/term/25
[7] http://tools.ietf.org/html/rfc2616
[8] http://tools.ietf.org/html/rfc2606
[9] https://sysportal.carnet.hr/taxonomy/term/22
[10] http://www.chiark.greenend.org.uk/~sgtatham/putty/