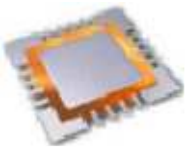

UNIVERZITET "Džemal Bijedić"
Fakultet Informacijskih Tehnologija
MOSTAR

Linux Command Shell

SEMINARSKI RAD

Mostar, Maj 2003

Student: **Nedim Hadžimahmutović**

Sadržaj

Uvod

Prva stvar s kojom se korisnici susreću na Linux operativnom sistemu je command shell ili komandno okruženje. Komandno okruženje je termin za tekstualno okruženje pomoću kojeg korisnik komunicira sa sistemom, koristeći se tastaturom (input) i monitorom (output). Korisnik instrukcije sistemu zadaje u vidu znakovnih komandi. Shell ima zadatak da prevodi korisnikove komande u instrukcije koje razumije operativni sistem. Moramo imati na umu da shell samostalno ne predstavlja Linux, nego samo njegovo korisničko sučelje, ili okruženje, koje može biti osim teksualnog i grafičkog karaktera. Shell je ustvari samo još jedan zaseban program, i kao takve prirode, izbor različitih shell programa je veliki. Najkorištenije komandno okruženje, je Bourne Again shell, ili kraće *bash*, koji je danas u Linux svijetu postao standard. Pored *bash* najkorišteniji su Korn shell (*ksh*), i "Tenex C shell" (*tcsh*). Ako niste sigurni koji command shell program koristi vaša distribucija, ukucajte sljedeće **echo $SHELL**.

Komandno okruženje predstavlja korisničko okruženje, koje posjeduje sve mogućnosti pomoću kojih možemo obavljati naše svakodnevne potrebe. Dostupni su nam programi za: komuniciranje putem e-mila, chata, instant messaginga, tekstualno procesiranje, održavanje sistema, i dr.

U ovom radu obrađen je bash shell, međutim ima govora i o drugim komandnim okruženjima. Iako je ovaj rad orijentisan Linux platformi, u nastavku rada u mnogim primjerima bit će spomenut UNIX. Tom prilikom trebamo imati na umu da je razvoj Linuxa bio cijelo vrijeme okrenut ka Unixu, odnosno glavni cilj je bio da se razvije besplatni i open source klon Unixa, te da se u mnogim slučajevima ova dva pojma ne mogu razdvajati jer imaju zajedničku historiju, sadašnjost a i budućnost.

Historija Komandnih Okruženja

Tokom razvoja Linux OS-a nastao je veliki broj komandnih okruženja, ali samo ih je nekoliko uspjelo da doživi rasprostranjenje na svjetskom planu.

Prvo važnije okruženje bilo je Bourne shell, koje je dobilo ime po svom originalnom kreatoru, Steven Bournu. Ovo okruženje je bilo uključeno u prvu popularnu verziju Unixa, Verziju 7 u 1979 godini. Bourne shell na sistemu je poznat kao *sh*. Pored svih promjena kroz koje je Unix

prošao, Bourne shell je još ostao popularan i gotovo ne promijenjen. Mnogi alati i administracijske funkcije se oslanjaju na ovo okruženje.

Prvo široko rasprostranjeno alternativno okruženje je bio C shell, ili csh. Ovo okruženje je napisao Bill Joy na Univerzitetu Kalifornije na Berkleyu kao dio Berkley Software Distribution (BSD) verzija UNIX sistema, koja je izašla nekoliko godina poslije Verzije 7. Ovo okruženje je uključeno u sve trenutne verzije UNIXa. C shell dobilo je ime po sličnosti komandi sa C programskim jezikom. Na ovaj način UNIX programeri mogu mnogo lakše savladati ovo komandno okruženje.

Osim dva predhodno navedena, okruženje koje je poslato popularno je Korn shell. Ovo okruženje je komercijalni produkt koji objedinjava najbolje sposobnosti Bourne i C shellova, plus mnoge sopstvene mogućnosti.

Bourne Again Shell

Bourne Again Shell okruženje dobilo je ime po Steve Bourneovom shellu, i kreirano je za korištenje u GNU projektu. Gnu projekat je započeo Richard Stallman, iz Free Software Fondation (FSF), sa ciljem kreacije UNIX kompatabilnog sistema i zamjene svih komercijalnih UNIX programa sa slobodno distributivnim programima. Na ovaj način nisamo dobili samo besplatne programe, već i novi koncept, tzv. *copyleft*. Copyleft je ustvari licnesa, a programi koji su napravljeni pod ovom licnesom mogu biti slobodno distribuirani, osim u slučaju kada je postavljena restrikcija na daljno distribuiranje (na primjer, izvorni kod programa mora biti slobodno dostupan).

Bash okruženje je razvijano da postane standardno komandno okruženje na GNU sistemu, i prvi put je obajvljeno u Nedjelju, 10 Januara, 1988 godine. Brian Fox je napisao originalnu verziju i nastavio rad i u sljedećoj dekadi. U ranoj 1989 godini, prodružio mu se Chet Ramey, koji je ovom okruženju doprinio u rješavanju mnogih greški i dodavanjem mnogih opcija. Chet Ramey je kasnije zamijenio Brian Foxa i postao vođa projekta. Prateći GNU principe, sve verzije bash okruženja, poslije 0.99 su slobodno dostupne iz FSF. Danas bash je standardno okruženje na Linuxu, i široko je korištena besplatna verzija UNIX operativnog sistema.

Korisničko Okruženje

Shell kao korisničko okruženje ima ogromne mogućnosti. Dostupne su velike skupine programa, od tekstualnih procesora, programa za komunikaciju, administraciju sistema, pa do raznih programerskih alata. Jedna od glavnih osobina komandnog okruženja jeste da ne zahtijeva velike hardverske resurse, i radi toga se odlikuju brzinom. Ta mogućnost nam pruža, da stare 384. 486 mašine budu ponovo od koristi.

Kada želimo da radimo u komandnom okruženju, potrebno je da se prvo prijavimo kao neki korisnik, unoseći svoje korisničko ime i lozinku. Nakon toga, korisnik ima mogućnost da pomoću komandi sistemu zadaje razne zadatke. Korisnik može da se odjavi kucajući komandu **exit** ili kombinacijom tipki CTRL+D.

Shell komande su linije sastavljene od jedne ili više riječi, koje su odvojene razmakom ili TABom. Prva riječ u liniji je *komanda*. Ostatak (ako ih ima) su *argumenti* (također zvani *parametri*) datoj komandi, a predstavljaju imena objekata na koje će komanda djelovati.

Kao primjer uzet ćemo komandnu liniju **lp datoteka**. Ova linija se sastoji od komande *lp* (printaj datoteku) i jednog argumenta **datoteka**. Komanda *lp* tretira **datoteka** kao ime datoteke za printanje. Iz ovom primjera možemo zaključiti da argumenti mogu biti imena datoteka, što ne predstavlja uslov. U komandnoj liniji **mail grungy**, mail program (komanda) tretira **grungy** kao korisnika, kome će mail biti poslan.

Komanda linija pored same komande i argumenta može da se sastoji i od tzv. *opcije*, koju šaljemo komandi. Opcija je specijalni tip agumenta koja daje komandi specifične informacije šta treba da radi. Opcije obično se sastoje od znaka crtice (-), iza koje dolazi slovo. Međutim, umjesto samo jednog slova, mogu doći dva ili više slova. Komanda **lp -s datoteka** sadrži opciju -s, koja govori *lp* programu da printanje izvrši u silent modu.

Rad Sa Datotekama i Direktorijima

U komandnom okruženju najčešće operacije koji Linux korisnik obavlja su razni vidovi rada sa datotekama. Kada kažemo datoteka, mislimo na sve što je vidljivo na sistemu, i sadrži neke vrste informacija. Postoje razne vrste datoteka, a najvažnije su:

Obične datoteke - ili tekstualne datoteke, su datoteke koje su sadrže čitljive karaktere.
Izvršne datoteke - također se zovu programi, kao npr. komande.
Direktoriji - sadrže druge datoteke i subdirektorije, odnosno njihova imena i druge informacije.

Najvažnija stvar kod direktorija je da mogu sadržati i druge direktorije (subdirektorij), što vodi do hijerarhijske strukture, odnosno *tree* strukture. Direktorij koji se nalazi u samom vrhu naziva se root, i on nema svoje ime na sistemu. Sve datoteke imaju ime koje pokazuje njihovu lokaciju relativnu u odnosu na *root* direktorij. Znak / razdvaja međusobno te direktorije. Ovaj način prikazivanja naziva se potpuni ili apsolutni način. Ako uzmemo za primjer datoteku koja se zove *bash.pdf* i koja se nalazi u direktoriju *knjige*, koje se opet nalaze u *home* direktoriju korisnika *tux*, apsolutna putanja do datoteke *bash.pdf* bi izgledala:

/home/tux/knjige/bash.pdf

Pored *absolutnih* putanja, postoji i *relativna* putanja datoteka, koja se odnosi na direktorije i datoteke na trenutnom direktoriju. U prošlom primjeru, ako se već nalazimo u direktoriju */home/tux* i želimo ući u direktorij knjige, možemo to jednostavno postići komandom ***cd knjige*** (relativna putanja do datoteke) umjesto kucanja **cd /home/tux/knjige** (apsolutna putanja).

Poslije prijave korisnika, trenutni, tekući ili radni direktorij, automatski postaje korisnikov *home* direktorij. Tako naprimjer */home/tux* je tipični home direktorij, u ovom slučaju korisnika tux.

U slučaju da se home direktorij ne nalazi kao u klasičnom slučaju, možemo koristiti znak tildu (~). Tako, umjesto /home/tux/knjige možemo pisati ~/knjige, što predstavlja apsolutnu putanju. Ako želimo da promijenimo trenutni direktorij, i pregledamo ostalo možemo koristiti komandu **cd**, a ako želimo vidjeti koji je trenutni direktorij koristimo komandu **pwd**. Ponvo se možemo vratiti u svoj home direktorij, pomoću komande **cd ~** odnosno **cd ~tux,** ili kao alternativu pomoću komande **cd $HOME**. Riječ $HOME predstavlja varijablu o ćemu će biti riječ kasnije. U sljedećoj tabeli navedeni su neki primjeri rada sa komandom cd, podrzumijevajući da je trenutni direktorij, korisnikov home direktorij:

Komanda	Novi Trenutni Direktorij
cd knjige	/home/tux/knjige
cd knjige/linux	/home/tux/knjige/linux
cd ~/knjige/linux	/home/tux/knjige/linux
cd /usr/bin	/usr/bin
cd ..	/home
cd ../korisnik	/home/korisnik
cd ~korisnik	/home/korisnik

Imena datoteka i Wildcard karakteri

Nekada, trebamo da pokrenemo jednu komandu na više datoteka u isto vrijeme. Jedan od uobičajenih primjera je komanda *ls* koja izlistava informacije o datotekama. U najjednostavnijoj formi, bez navedenih argumenata, ova komanda daje listu svih datoteka u tekućem direktoriju osim skrivenih datoteka, odnosno onih datoteka čije ime počinje sa znakom tačke (.).

Ako komandi ls zadate kao argumenat ime neke datoteke, komanda samo prikaže u komandnoj liniji ime datoteke koje ste naveli. Ovo isto važi ako umjesto jedne datoteke, navedete dvije, ili više datoteka. Međutim, način na koji se komanda ls koristi je sa opcijama, kao što je **-l** (long) opcija, koja komandi govori da prikaže vlasnika, veličinu, vrijeme zadnje izmjene, i druge informacije o datoteci. Ako želimo vidjeti sakrivene datoteke, koristimo opciju **-a** (all). Međutim, nekada želimo da vidimo imena određene grupe datoteka, a da ne znamo njihova imena, npr. ako želimo vidjeti sve tekstualne datoteke, odnosno one čija se imena završavaju sa *.txt* završetkom (ekstenzijom). Ovo postižemo pomoću *wildcard* karaktera. Lista osnovnih wildcard karaktera:

Wildcard Karakter	Jednak sa
?	Samo jednim karakterom
*****	Bilo kojim stringom karaktera

Upotrebu wildcard karaktera ?, najbolje možemo shvatiti kroz primjer. Ako imamo u istom direktoriju datoteke *program.c*, *program.log* i *program.o* komanda **ls program**.? će izlistati samo imena *program.c* i *program.o*, jer se njihove ekstenzije sastoje iz samo jednog karaktera.

Wildcard karakter * je mnogo moćniji, i svakodnevno ga koriste svi korisnici komandne linije. Tako ako shellu zadamo komandu **ls program.*** kao odgovor dobit ćemo sve tri imena datoteka koja počinju sa skupom karaktera *program*, odnosno riječju. Također, ako želimo da vidimo sve

tekstualne datoteke u tekućem direktoriju, kucamo **ls *.txt**.

Redirekcija i Piping

Mogućnost korištenja pipinga je jedna od moći komandne linije. Korištenje ove tehnike u terminalu je dobar trik, a i predstavlja olakšanje pri radu. Piping je također široko raširena metoda među C programerima. Bez pipinga, bilo bi potrebno pisati velike količine koda, da bi se uradili jednostavni zadaci.

U osnovi Piping čini proces pokretanja neke komande i zatim slanje njenog izlazanog rezultata na drugi program ili datoteku, znači zapisivanje u tu datoteku.

Zamislimo da imamo tekst datoteku u tekućem direktoriju koja sadržati mnogo linija teksta sačinjenih od riječi TUX, a zatim jednu rečenicu koja ce sadržati riječ TUX i riječ Linux. Zatim ukucajte sljedeće:

grep TUX moja_datoteka.txt

Rezultat ovog zadataka će kao i obično biti izlistan na ekranu, a sačinjavat će pretragu za rečenicama koje sadrze riječ TUX. Međutim, ako ukucamo:

grep TUX moja_datoteka.txt > tux_recenice.txt

Rezultat možete vidjeti u datoteci tux_recenice.txt. Vidimo da je rezultat komande grep redirektiran i zapisan u datoteku tux_recenice.txt. Dio komande "**> tux_recenice.txt**" govori shellu da napravi novu datoteku pod nazivom tux_recenice.txt i u nju zapiše rezultat komande grep, umjesto da sve izlista na ekran. Ako već postoji datoteka sa istim imenom onda će rezultat komande grep biti zapisan preko sadržaja te datoteke.

Međutim, ako umjesto znaka **>** stavimo znak **>>**, sadržaj već postojeće datoteke će ostati isti, a rezultat će biti dodan na zadjnu liniju teksta. Pomoću **echo** komande možemo dodavati tekst u neku datoteku, npr.

echo "pingvin" >> tux_recenice.txt

Prava moć pipinga je kada jedan program može da čita output nekog drugog programa.

Posmatrajmo grep komandu kada je pokrenemo bez argumenata:

grep Tux
recenica koja sadrzi rijec Tux
recenica koja sadrzi rijec Tux
recenica bez te rijeci
Tux je pingvin
Tux je pingvin

grep čita sa ulaza, tastature, i ispisuje rečenicu u kojoj se nalazi riječ Tux, tako da takve rečenice su prikazane dva puta. Sada ukucajte

grep TUX moja_datoteka.txt | grep Linux

Prvi grep izlista sve Tux rečenice kao output. Znak | vrijednosti iz outputa prenosi na input sljedeću komandu, koja je takodjer grep komanda. Druga grep komanda pregleda sadržaj inputa u potrazi za rečenicama koje sadrže riječ Linux u sebi. Komanda grep na ovaj način se često koristi kao filter, i može se koristiti u više primjeraka, npr.

grep L moja_datoteka.txt | grep i | grep n | grep u | grep x

Također jedna od korisnih opcija je i znak <. Ovaj znak ima funkciju da redirektira sadržaj neke datoteke na input (stdin), odnosno sadržaj određene datoteke bi zamjenili ono što bi inače dolazilo sa tastature.
Primjer:
grep Tux < tux_recenice.txt

Sljedeći primjer je od velike koristi za administratore:

ps -aux | grep "ime-korisnika"

Komanda **ps -aux** izlistava sve procese (programe) koji su pokrenuti u tom momentu. Lista procesa će biti poslana na grep komandu, koja će taj sadržaj pretražiti u potrazi za datim stringom, u ovom slučaju "ime-korisnika". Krajnji rezultat na ekranu bi bila lsita svih procesa koji

je određeni korisnik pokrenuo. String "ime-korisnika" možemo zamjeniti sa bilo kojim imenom korisnika na toj linux mašini. Na ovaj način možemo pratiti aktivnosti određenih korisnika.

Opće Korištene Komande

Iako je do sada je bilo govora o komandama, u sljedećoj listi možete naći kompletniju listu komandi:

Gašenje i startanje kompjutera:

shutdown -h now	gasi sistem u istom momentu
shutdown -r 5	gasi sistem na 5 minuta, a zatim ga reboota
shutdown -r now	gasi sistem a zatimga reboota
halt	gasi kompjuter - isto kao shutdown -h now
startx	starta X windows

Pronalaženje datoteka

find / -name datoteka	traži ime datoteke "datoteka počevsi iz root direktorija
find / -name "*ime*"	traži datoteku koji u sebi ima 'ime' počevši iz root direktorija

Pomjeranje, kopiranje, brisanje & pregled datoteka:

ls -l	izlistava datoteke u tekućem diru koristeći long opciju
ls -F	izlistava datoteke u tekućem diru i pri tome označava vrstu datoteka
rm ime	briše datoteku ili direktorij imena 'ime'
rm -rf ime	briše cijeli direktorij, sve datoteke i poddirektorije u njemu
cp datoteka /dir	kopira datoteku u /dir
mv datoteka /dir	premješta datoteku imena datoteka u direktorij /dir
cat datoteka	prikazuje sadžaj navedene datoteke
man imekomande	prikazuje man stranicu navedene komande
more datoteka	prikazuje datoteku jednu po jednu stranicu
head datoteka	prikazuje 10 prvih linija datoteke
head -20 datoteka	prikazuje prvih 20 linija datoteke
tail datoteka	prikazuje zadnjih 10 datoteke
tail -20 datoteka	prikazuje zadnjih 20 linija datoteke

User Administracija:

adduser korisnik	dodavanje novog korisnika
passwd korisnik	dodjeljivanje passworda korisniku
su	logiranje kao super korisnik, administrator odnosno root korisnik
exit	odjava trenutno prijavljenog korisnika

Tekst Editori:

vi	text editor koji je standard na svim Linux distribucijama
pico	također jedan dobar tekst editor
emacs	*kreacija Richarda Stallmana - napredan editor*

Prilagođavanje Korisničkog Okruženja

Okruženjem možemo smatrati kolekciju koncepata koji predstavljaju stvari koje sistem, ili neki drugi skup alata rade, u smislu stvaranja okruženja čiji će izgled davati korisniku osjećaj komfornosti. Okruženje možemo porediti s radnim stolom gdje radimo. Na stolu se nalaze razni predmeti za svakodnevnu upotrebu, kao što su: olovke, diskete, CDovi, knjige, papiri, koverte, telefon, kalkulator i dr. Svaki radnik stol prilagođava svojim potrebama, i grupiše predmete po njihovim karakteristikama. Olovke stavlja na mjesto gdje ih može dohvatiti rukom, također su i najčešće korišteni CDovi nadohvat ruke i sl. Što je prilagođavanje radne okoline ličnim potrebama veće, zadovoljniji i organizovaniji ste, i time vaša produktivnost se povećava.

Slično ovome, Linux vam pruža mnoge programe i mogućnosti da svoje radno okruženje prilagodite svojim ličnim potrebama, manipulacijom datoteka, direktorijia, komandi, izgleda u vidu standardnog outputa i inputa. Bash pruža četiri glavna načina prilagođavanja radnog okruženja. Ti načini su:

Specijalne datoteke

Datoteke *.bash_profile*, *.bash_logout*, i *.bashrc* su datoteke koje čita bash prilikom prijave, odjave korisnika i pokretanja novog shella.

Aliasi

Sinonimi za komande ili komandne stringove koji se mogu definisati.

Opcije

Kontrole za različite aspekte okruženja, koji se mogu uključiti i isključiti.

Varijable

Promjenljive Vrijednosti koje preporučuju nešto pomoću imena.

.bash_profile*, *.bash_logout*, i *.bashrc datoteke

Ove datoteke za *bash* imaju posebno značenje, jer omogućavaju način korisniku da postavi odgovarajuće okruženej automatski prilikom samog prijavljivanja, ili prilikom pozivanja drugog bash shella, također pozivanja komandi prilikom napuštanja shella. Ove datoteke inače postoje u vašem home direktoriju, a ako ih nema onda se koristi globalna datoteka za sve korisnike /etc/profile, zavisno od sistem administratora. U slučaju da ove datoteke ne postoje, možete ih sami napraviti.

Najvažniju datoteku *.bash_profile*, bash čita i komande u njoj izvršava svaki put prilikom prijave korisnika. Ako pogledamo sadržaj *.bash_profile* datoteke vidjet ćemo nešto slično ovome:

```
PATH=/sbin:/usr/sbin:/bin:/usr/bin:/usr/local/bin
SHELL=/bin/bash
MANPATH=/usr/man:/usr/X11/man
EDITOR=/usr/bin/vi
PS1='\h:\w$ '
PS2='> '
export EDITOR
```

Ove linije definišu osnove za korisnikovo okruženje. Ako ste početnik najbolje je da ih ne mijenjate, već ispod postojećih dodajete nove komande. Međutim, morate imati na umu da promjene koje ste napravili neće biti primjenjeni sve dok *.bash_profile* datoteka nije ponovo pročitana nakon odjavljivanja, pa ponovnog prijavljivanja korisnika. Kao alternativno riješenje može poslužiti komanda **source** koja izvršava komande navedene u nekoj datoteci, primjer:

```
source .bash_profile
```

Na ovaj način proces odjavljivanja i ponovnog prijavljivanja istog koriniska je nepotreban.

Bash dozvoljava dva sinonima za ovu datoteku: *.bash_login* i *.profile*. Samo jedna od ove tri datoteke će biti čitana prilikom prijave korinsika. Ako *.bash_profile* ne postoji u home direktoriju *bash* traži *.bash_login* datoteku, a ako i ona ne postoji na red dolazi *.profile* datoteka.

Datoteka *.bash_profile* se čita samo prilikom prijavljivanja korisnika. Ako pokrenemo novi shell (subshell) kucanjem komande **bash** u komandnoj liniji, komande u datoteci *.bashrc* će biti pročitane. Na ovaj način imamo fleksibilnost odvajanja komandi koje su potrebe prilikom startupa i onih potrebnih prilikom pokretanja subshella.

Datoteka *.bash_logout* je čitana i izvršena svaki put kada korisnik prilikom odjave napušta shell. Ova datoteka je veoma korisna, ako želimo da vratimo promijenjene postavke u

komandnom okruženju, ili da pokrenemo komande koje će izbrisati temporary datoteke, ili zabilježiti koliko dugo je korisnik ostao prijavljen na sistemu. Ova datoteka se ne mora nazaliti u korisnikovom home direktoriju i u tom slučaju dodatne komande prilikom napuštanja shella nisu izvršavane.

Aliasi

Prilikom rada na Linux sistemu često ćete se susresti sa komandama koje imaju komplikovane nazive. Ponekada komande koje se često koriste sadrže mnogo opcija i argumenata koji se moraju navesti. U ovom slučaju radi pomoći *bash* nam omogućuje *aliase*. Pomoću aliasa, dugačke komande ili komande sa puno argumenata i opcija možemo skratiti, ili ima dati totalno druga imena. Na ovaj način komandama možemo dati jednostavnija imena, ili čak ih prevesti na maternji jezik.

Aliasi se mogu definisati u komandnoj liniji, u .bash_profile ili .bashrc datoteci, koristeći ovu formu:

alias novoime=komanda

Iz navedene sintakse možemo zaključiti da *novoime* predstavlja alias za komandu po imenu komanda. Svaki put kada kucate *novoime* kao komandu bash će izvršiti komandu *komanda*. Jedan od pravila sintakse je da ne smije biti praznih mjesta (space) ni na jednoj strani znaka jednako (=).

Postoji nekoliko osnovnih načina na koje možete koristiti aliase.

Neki ljudi koji ne kucaju dobro, i prave često iste greške prilikom kucanja istih riječi, mogu koristiti aliase da umanje te greške. Primjer:

alias zast=yast
alias mali=mail
alias gerp=grep

Iako ovo može biti od koristi preporučeno je da se korisnici suoče sa ovim problemom i savladaju slične pravopisne greške.

Jedan od uobičajenih načina korištenja aliasa je, ako u svakodnevnom poslu trebate da

često pristupate direktoriju koji je zakopan duboko u direktorijskoj hijerarhiji. Sljedeći primjer pokazuje kako riješiti ovaj problem tako da ne terbamo tipkati putanju do tog direktorija, a čak je ni pamtiti:

alias cdsem='cd Documents/fakultet/radovi/seminarski'

Znakove navodnika moramo staviti oko **cd** komande ako se ta linija sastoji više od jedne riječi. Sada je dovoljno da ukucamo **cd $cdsem** da pristupimo *Documents/fakultet/radovi/seminarski* direktoriju.

Kao dodatak, razmotrit ćemo osnovne načine korištenja same komande **alias**. Ako kucamo komandu **alias** *ime* bez znaka jednakosti (=), shell će nam pokazati vrijednost tog aliasa, ili u slučaju da taj alias nije definisan dobit ćemo **alias** *ime* **not found**. Ako kucamo u shell **alias** bez ikakvih argumenata, dobit ćemo listu svih aliasa koji si definisani. Komanda **unalias** *ime* uklanja definisani alias argumenta, odnosno *ime* alias.

Aliasi su veoma dobro rješenje za korisnike koji su početnici sa Linuxom. Danas, većina naprednijih korisnika umjesto aliasa koristi razne shell skripte i funkcije, koji mogu uraditi sve što i aliasi, čak i mnogo više, međutim za njih je potrebno znanje programiranja.

Opcije

Aliasi mogu pomoći korisniku da promijeni ime komande, ali ne i samo ponašanje okruženja. Opcije predstavljaju jedan način da ovo uradite. Shell opciju možemo smatrati kao postavku koja je "on" ili "off", odnosno uključena ili isključena. Opcije se rasprostiru od onih koji su samo od interesa za programera, pa da interesa običnih korinsika. U ovom radu govorit ćemo o opcijama od kojih sve vrste korisnika mogu imati koristi.

Najjednostavniji oblik komande koje predstavljaju *opcije* su **set -o** *imeopcije* i **set +o** *imeopcije*. Znak **-** isključuje navedenu opciju, dok znak **+** uključuje navedenu opciju. Imena opcija se mogu sastojati i samo od jednog slova. Tako na primjer **set -o noglob** može se također napisati i kao **set -f**. Lista osnovnih shell opcija:

Opcije	Opis
emacs	Emacs način editovanja.
ignoreeof	Zabrana korištenja CTRL+D za napuštanje shell okruženja, samo je dozvoljena komanda exit za napuštanje okruženja
noclobber	Zabrana output redirekcija (>) za prepisivanje preko postojeće datoteke
noglob	Zabrana produženja imena datotekama wildcard znakovima kao što su * i ?
nounset	Prikazuje se error poruka prilikom korištenja varijable koja nije definisana, odnosno nedefinisane varijable ne tretira kao nulte
vi	Vi način editovanja
notify	Shell odmah šalje report prilikom ukidanje pozadiniskih poslova (background jobs)
history	Omogućavanje bilježenje historije komandi
monitor	Omogućavanje kontrole poslova
privileged	Skripta se pokreće u suid modu
onecmd	Napust shell poslije čitanja i pokretanja jedne komande
verbose	Printaj shell input linije prije njihovog pokrećanja

Varijable

Shell varijable imaju mogućnost da sve karakteristike korisnikovog okruženja definišemo unutar verijabli. Varijable mogu primiti razne vrijednosti, od raznih shell stringova do koliko shell često provjerava mail.

Kao i opcija, shell varijabla predstavlja ime koje sadrži određenu vrijednost. *Bash* sadrži mnoge ugrađene varjable, dok programeri, ili napredni korisnici mogu dodavati svoje proizvoljne varijable. Sintaksa varijabli je slična kao i za aliase:

varijablaime=vrijednost

Ne smije biti praznih mjesta ni s jedne strane znaka jednakosti, i ako je vrijednost više od jedne riječi, mora biti okružena znakovima navodnika (' '). Ako želimo koristiti neku od varijabli u komandnoj liniji moramo prije imena varijable staviti znak dolara ($).

Ako želimo ukloniti varijablu koristimo se komandom **unset** *varime*. Ovo može biti od koristi u slučaju da koristimo opciju **nounset**, gdje prilikom korištenja nedefinisane varijable shell prikazuje error poruku, a inače u normalnom shell okruženju, nedefinisane varijable imaju nultu vrijednost i shell ignoriše korištenje nedefinisanih varijabli.

Najlakši način da se provjeri vrijednost varijable je da se koristi ugrađena komanda **echo**. Komanda echo služi za printanje njenih argumenata. Iz ovoga možemo zaključiti, ako varijabla **GNU** ima vrijednost **GNU nije UNIX**, kucajući:

echo "$GNU"

u komandnoj liniji dobit ćemo odgovor **GNU nije UNIX**. Ako navedena varijabla nije definisana onda shell printa prazno mjesto. Isti rezultat možemo postići i na kompleksniji način:

echo "Vrijednost **$**varijablaime **varijable je** **"$**varijablaime**"."**

Kao što već znamo, znak \ govori bash shellu da prvi karakter iza njega ignoriše. U primjeru vidimo da se prvi dolar znak, nalazi između duplih znakova navodnika (""), i da se nalazi poslije znaka \ (backslash-escaped) tako da se ignoriše. Inače to nije slučaj ako se nalazi samo između znakova navodnika, pošto znak $ spada u grupu specijalnih karaktera.

Ugrađene Varijable

Kao i kod opcija, neke ugrađene shell varijable su od posebnog značaja Linux korisnicima, dok su druge od koristi jedino hackerima. Mi ćemo se osvrnuti na one koje generalno koristi većina korisnika. Lista korisnih ugrađenih varijabli:

Ime Varijable	Opis
HISTCMD	Historija broja ukucanih komandi
HISTCONTROL	Prikazuje opciju koji je uključen za bilježenje histrije ukucanih komandi. Opcije mogu biti ignorespace - ne bilježe se linije koje počinju sa razmakom, ignoredups - ne bilježe se komande koje su iste kao predtohno ukucane, i ignoreboth - uključuje obe navedene opcije
HISTFILE	Datoteka u koju se bilježi historija ukucanih komandi. Ta
HISTFILESIZE	Maksimalan broj komandi za čuvanje u historijskoj datoteci
FCEDIT	Putanja do editora za korištenje sa fc komandom
BASH	Puna putanja do bash shella.
EUID ili **UID**	Prikazuje User ID za trenutnog korinsika
HOME	Kućni (home) direktorij trenutnog korisnika
HOSTNAME	Ime host mašine
HOSTTYPE	Tip mašine - arihtektira kompjutera
INPUTRC	Datoteke čije se linije čitaju prilikom startanja kompjutera
LANG	Lokalni jezik
SECONDS	Broj sekundi odkako je korisnik prijavljen
PATH	Put za traženje komandi - programa
USER	Ime korinsika

Vrijednosti svih navedenih varijabli možemo vidjeti pomoću komande echo:

echo $IMEVARIJABLE

Prompting Varijable

Pod pojmom prompt mislimo na kursor koji trepti na vašoj komandnoj liniji i riječi prije njega, koji npr. pokazuju vaše korisničko ime, ime host računara i druge podatke. Ti podaci se proizvoljno mogu mijenjati, pomoću prompt stringova. Ovi stringovi su smješteni u varijablama **PS1**, **PS2**, **PS3**, i **PS4**. Prvi od ovih stringova se zove primary prompt string, koji je inače i uobičajeni shell prompt. Njegova uobičajena vrijednost je "\s-\v\$". Međutim, većina distribucija vrijednost ovo stringa izmjenjuje. Sljedeći primjer pokazuje kako promjeniti da prompt pokazuje vaše korisničko ime:

PS1="\u--> "

Znakovi \u govori bash shellu dad ubaci ime trenutnog korisnika u prompt sting. Ako u prompt

sting stavimo i \! vrijednost, hostorija komandi će biti prikazan. Vrijednost \w nam u promptu prikazuje trenutni direktorij u kome se nalazimo, čime dobijamo na vremenu jer ne trebamo neprestalno kucati **pwd** komandu. Lista vrijednosti za podešavanje prompta:

Komanda	Značenje
\a	ASCII bell karakter (007).
\d	datum u formatu "Radni Dan Mjesec Dan"
\e	ASCII escape karakter (003)
\H	ime host računara
\h	ime host računara do "."
\n	carriage retun i line feed
\s	ime okruženja
\T	trenutno vrijeme u 12-sati HH-MM-SS
\t	trenutno vrijeme u formatu HH-MM-SS
\@	trenutno vrijeme u formatu 12-sati am/pm
\u	korisničko ime trenutnog korisnika
\v	verzija bash-a (npr. 2.00)
\V	izdanje bash-a; verzija i patchlevel (npr. 2.00.0)
\w	trenutni radni direktorij
\W	ime trenutnog radnog direktorija
\#	ime komande trenutne komande
\!	historija broja trenutne komande
\$	ako je efektivni UID nula, printat će #, u suprotnom printat će $
\nnn	oktal je karakter u code-u
\\	printaj backslash
\[	započni sekvenciju od ne isprintanih karaktera, kao terminalne kontrolne sekvencije
\]	završi sekvenciju od ne isprintanih karaktera

Ako izvršimo promjene u **PS1** stringu, one će važiti samo dok smo logovani kao taj korisnik, odnosno prestaju važiti ako se korisnik odjavi ili pokrene subshell. Ako želimo da sačuvamo trenutni promjene koje smo učinili sa izgledom našed prompta, kucaćemo sljedeću komandu:

SAVE=$PS1

PS2 se zove sekundarni prompt sting, i njegova uobičajena vrijednost je >. Koristi se u slučaju kada poslije nedovršene riječi neke komande pritisnemo Enter, kao napomena da moramo dovršiti komandu. Na primjer, kada kucamo komandu sa navodnicima, i ne zatvorimo navodnike, pojavit će nam se znak > u sljedećem redu, što je znak da trebamo da zatvorimo navodnike. Primjer možemo vidjeti ovdje:

echo "Ovo je primjer,
>koji se zavrsava ovdje"

ZAKLJUČAK

U ovom radu pokušao sam da objasnim osnove korištenja komandne linije, načine podešavanja izgleda okruženja, i druge teme koje su iznad nivoa početnika. Jedan od ciljeva mi je bio da pokažem važnost komandog okruženja, i njegovu moć. Naravno, da sam nastavio s tom idejom, i da nisam imao nekih ograničenja kao što je vrijeme, ovaj rad bi se protezao od shell programiranja do administracije sistema i procesa, a 20 stranica bi bilo samo prvo poglavlje. Kao dugogodišnji korisnik bash shella, mogu reći da mi je u mnogim slučajevima mnogo bilo lakše raditi, i snaći se u shellu, nego u GUI okruženju. Mnoge sam tekstove izkucao u konzoli, chatao, slao mailove, i opet sam ovaj način prihvatio kao interesantniji. Ako gledamo sa principa učenja programiranja, smatram da se najbolje savlada pravi koncept programiranja u shellu, a ne iz vizuelnih alata. Bacite li koji pogled po domaćim stranicama primjetit ćete da iz dana u dan sve više ima zainteresovanih za bash i Linux generalno, a rad na programima se i dalje nastavlja. Čak se pristupa načinu programiranja gdje se kombinuju konsolne i GUI aplikacije. Pored što se prave GUI interfejsi za već postojeće konsolne programe, aplikacije koje su pravljene samo za GUI, sve više se sastoje iz konsolnih programa sa specifičnom namjenom. Jedna od prednosti ovoga pristupa jeste da te programe mogu koristiti, odnosno s njima komunicirati i druge aplikacije u vidu komandnih instrukcija. Sa gledišta programera, ili korisnika koji treba da uradi manje programerske zadatke, a nedostaje mu znanje programiranja, bash shell je odlično okruženje za tu svrhu. Postoji ogroman broj distribuiranih programa, za sortiranje karaktera, datoteka, njihovo pretraživanje, upoređivanje, spajanje, dijeljenje, zatim već prenesenih funkcija iz C jezika itd. I za kraj, ako naiđete na suprotno mišljenje sjetite se "stare" Linuxaške izreke:

Bash je baš logičan.

LITERATURA

1. ***Learning The Bash Shell**, Second Edition*

   Cameron Newham and Bill Rosenblatt - O'REILLY

2. ***Informacijska Tehnologija**, Drugo Izdanje*

   Dr Nijaz Bajgorić

3. ***URLS:***

   http://dokumentacija.lugze.org

   http://howto.linux.org.ba