AIOS2 01 - Praćenje i upravljanje procesima

- **Proces** je program koji se izvršava.
- Sastoji se od adresnog prostora alocirane memorije, niti, stanja procesa i bezbednosnih svojstava.
- Okruženje procesa obuhvata varijable, kontekst i alocirane sistemske resurse.
- Roditeljski proces duplira svoj adresni prostor (fork operacija) da kreira strukturu novog (child) podpro.
- Svaki proces ima svoj **PID** i svaki može kreirati podproces. Svi procesi su potomci prvog proc. - systemd.
- Fork rutinom podproces nasleđuje podatke i programski kod roditelja a potom pokreće sopstveni kod.
- Završetkom podproc. odbacuje sve korišćene resurse, šalje roditelju exit signal i prelazi u zombi stanje.

Stanja procesa:

- Running

R - TASK_RUNNING. Proces se ili izvršava ili čeka na izvršavanje, učitan je u memoriju.

- Sleeping

S D K - TASK_INTERRUPTIBLE/UNINTERRUPTIBLE/KILLABLE. Proces čeka neki uslov da zadovolji, kada se zadovolji menja stanje u running. / Ne šalje odgovor na bilo kakav signal. / Slično prethodnom, samo što dopušta primanje signala za gašenje.

- Stopped

T - TASK_STOPPED/TRACED. Proces je zaustavljen obično na signal korisnika ili drugog procesa, može biti nastavljen. / Proces se debaguje.

- Zombie

Z X - EXIT_ZOMBIE/DEAD. Podproces signalizira roditelju prekid rada. Svi resursi osim PID-a su otpušteni. / Kada roditeljski proces očisti SVE strukture podataka podprocesa, tada je on u potpunosti završen.

- Job Control je shell svojstvo koje omogućava pokretanje više komandi istovremeno.
- Proces u prvom planu (FG) je trenutno aktivan u terminalu, prima ulaz sa tastature i signale. Samo 1.
- Procesi u pozadini u programu ps kod kolone TTY imaju ?. Pokreću se sa & na kraju komande.
- jobs daje listu svih procesa sa Job ID-jem, fg /bg %1 pomera proces sa JID-jem 1. ps j sve pozadinske lista.

Prekidni signali procesa:

- Prekid rada foreground procesa se radi pomoću tastera **Ctrl+C** (**Signal 2, keyboard interrupt**)
- **Ctrl+** proizvodi prekid procesa ali i njegov dump (snimanje stanja u datoteku)(**Signal 3, keyboard quit**)
- **Signal 9** (**kill, unblockable**), ubija proces odmah bez davanja vremena procesu. Nasilan prekid procesa.
- Prekidni **signal 15** je podrazumevani signal većine signala (**TERM, terminate**), regularno prekida program, kada ga pokrenemo program se gasi ali puštamo programu da prekine sve akcije koje su u vezi sa njegovim resursima.
- **CTRL+Z** je **Keyboard stop** (**signal 20**), proces se zaustavlja ali može biti nastavljen (jos uvek je učitan u memoriju samo njegovi threadovi nisu učitani u memoriju)

- **Uptime** pokazuje opterećenje svih procesora zajedno u poslednjih 1, 5 i 15 minuta. Za pojedinačni procesor moramo sve 3 vrednosti podeliti sa brojem procesora/jezgara.
15:11:13 up 2:11, 3 users, load average: 2.95, 4.40, 5.20 - Ako ima 3 jezgra onda je za 1 min. 0.98.
1.0 = 100% iskorišćenje, 1.3 = Postoji 30% više threadova koje CPU treba da izvrši, 0.73 = Procesor ne koristi 27% svog vremena. Load average se uvećava kada zahtevi (niti raznih procesa) čekaju u redu CPU.

AIOS2 02 - Dnevnici događaja i NTP

- Datoteke sistemskih dnevnika događaja (servisi **systemd-journald** i **rsyslogd**) se nalaze u /var/log.
* /**var/log/messages** : Najveći broj poruka se snima ovde, osim onih koje se snimaju u druge datoteke.
- var/log :
* **/secure** : Bezbednosne i autentifikacione poruke.
* **/maillog** : Poruke u vezi rada servera elektronske pošte.
* **/cron** : U vezi zadataka koji se periodično pokreću.
* **/boot.log** : U vezi procesa podizanja operativnog sistema.

- Pomoću logger-a šaljemo poruke sistemu dnevnika. logger “poruka” -> Slanje poruke u messages.

Kako se događaji snimaju oni mogu biti sledećih tipova, 7 je najniži a 0 najvišeg prioriteta:
kod - prioritet - ozbiljnost poruke
* 0 - emerg - sistem je nestabilan
* 1 - alert - potrebna trenutna akcija
* 2 - crit - kritično stanje
* 3 - err - stanje nekritične greške
* 4 - warning - upozorenje na događaj
* 5 - notice - uobičajen ali značajan događaj
* 6 - info - informacioni događaj
* 7 - debug - poruka nivoa ispravljanja grešaka (Debugging Level message)

- U žurnalu se privremeno zadržavaju podaci, on sadrži originalne nesortirane poruke.
- Rotacija datoteka se izvodi pomoću **logrotate** koji se startuje preko **cron** servisa sa ciljem brisanja zastarelih datoteka dnevnika. Datoteka iz /var/log/messages biva rotirana u -ll--20160610 i stvara se nova datoteka, a posle nekog vremena ta rotirana datoteka se briše.

Format događaja:
1. Datum i vreme kada se događaj desio,
2. Računar koji je poslao poruku,
3. Program ili proces koji je poslao poruku,
4. Poruka.

1 2 3 4
Feb 25 12:23:04 localhost su: pam_unix(su-l:session) : session opened for user root.

Rad žurnala
- Servis systemd-journald snima informacije o događajima u žurnal datoteku. Ovde se snimaju podaci o događaju npr. njegov tip i prioritet. Pristupa mu se naredbom **journalctl** (-n 5 -p 3 --since today npr.)
- Nalazi se u **/run/log/journal**. Pregled žurnala se radi pomocu komande journalctl.
- Pošto se žurnal briše svaki put kod restarta, procedura je sledeća. mkdir -p /var/log/journal, chown root:systemd-journal/var/log/journal, chmod 2755 /var/log/journal i killall -USR1 systemd-journald.

- Pomoću **Network Time Protocol - NTP** računar sinhronizuje vreme sa javnim NTP serverom na internetu ili loklanim NTP serverom. chronyd je default NTP klijent/server (chronyc je klijent).
- U /etc/chrony.conf, **iburst** omogćava da chronyd servis izvrši 4 merenja radi preciznije sinhro.
- tzselect nam postavlja pitanja i tako na kraju odredi vremensku zonu.
- timedatectl set-ntp true/false : Uključivanje/isključivanje NTP sinhronizacije.
- timedatectl set-time 15:35:00 => Podesimo vreme na 15:35
- chronyc sources -v => Vidimo servere odakle se prikupljaju informacije o vremenu.

AIOS2 03 - Bash skripte

- Programi se dele na kompajlerske i interpreterske. Skripta je interpreterski program.
- **Kompajlerski program** se prevodi na mašinski jezik u trenutku kompajliranja programa.
- **Interpreterski** zahtevaju postojanje posebnog programa koji ih prevodi na m. jezik u realnom vremenu.
- Svaka skripta treba da započne stringom #!/bin/bash - **sha-bang**.
- Nakon pisanja skripte i snimanja datoteke, treba dodeliti execute ovlašćenje tj. **chmod 774 <skripta>**.
- Skripta se pokreće koršićenjem relativne putanje ./<skripta> ili apsolutne putanje.
- # = Komentar
- \ = Uklanja značenje nekog specijlanog karaktera tipa #
- Tekst smešten između 2 apostrofa '' se interpretira na bukvalan način.
- Tekst izmedju navodnika "" zadržava bukvalno značenje osim znaka $, \, i backtick karaktera (`).
- Tekst smešten između 2 backtick karaktera `` se izvršava pre glavne komande.
- Supstitucija korišćenjem dva backticka je zastareo način. Moderni način je korišćenjem $ karaktera.
- Varijable omogućavaju čuvanje neke vrednosti u memoriji. Sintaksa: naziv varijable=vrednost.
- Vrednost varijable se može dobaviti ekspanzijom varijable. $varijabla => echo $ime.
- Preinkrement operator prvo uveća varijablu za 1 pa je koristi, dok postinkrement operator prvo koristi varijable pa je onda uveća za jedan.

Petlja for

Sintaksa:

```
for <VARIABLE> in <LIST>
do
<COMMAND>
...
<COMMAND>
done
```

```
for i in 1 2 3 ili for i in {1..3} ili for FILE in /etc/*
do
echo “Vrednost je $i” ili echo “Ime fajla je $FILE”
done
```

- Petlja procesira elemente u listi <LIST>, jedan po jedan i završava se nakon procesiranja poslednjeg elementa.

Aktivacija **debagovanja** se radi na 2 načina:

1. Dodavanjem opcije -x na kraju prvog reda skripte, #!\bin\bash -x,
2. Pokretanjem skripte sa -x opcijom, bash -x <skripta>

- U debug režimu ispisuje se komanda neposredno pre njenog izvršavanja.
- Moguće je uključiti debagovanje za samo deo neke skripte. Uključuje se sa set -x, a isključuje se komandom set +x.

AIOS2 04 - Uslovni izrazi i kontrolne strukture

- Pozicioni parametri su varijable koje sadrže vrednost argumenata skripte. $* sve 1 red; $@ novi red.
* Napravimo skriptu cp $1 $2, pa probamo ./script file1 folder1. $# nam govori broj poz. par. (ovde 2).
- Svaka komanda vraća izlazni kod. Uspešno završena komanda **vraća 0**, neuspešna komanda broj različit od 0. Proverava se sa **echo $?** .
- Komanda exit omogućava prekid izvršavanja skripte.

```
- Poredjenje se vrsi koriscenjem izraza oblika: [ <vrednost1> <operator> <vrednost2> ]
- echo $? => 0 znaci da je poredjenje uspesno, sve ostalo da je neuspesno.
- Logicki operatori AND (&&) i OR (||). [ USLOV1 ] ||/&& [ USLOV2 ]
Operatori:
-eq : Jednako            [ "$a" -eq "$b" ]   = : Jednako          [ "$a" = "$b" ]     -z : String je null      [ -z "$b" ]
-ne : Nije jednako       [ "$a" -ne "$b" ]   == : Jednako         [ "$a" == "$b" ]    -n : String nije null    [ -n "$b" ]
-gt : Vece od            [ "$a" -gt "$b" ]   = : Nije jednako     [ "$a" != "$b" ]    -d : Direktorijum postoji [ -d <DIRECTORY> ]
-ge : Vece ili jednako   [ "$a" -ge "$b" ]                                            -e : Datoteka postoji     [ -e <FILE> ]
-lt : Manje od           [ "$a" -lt "$b" ]                                            -r -w -x : Datoteka ima ovlascenja [ -r/w/x <FILE> ]
-le : Manje ili jednako  [ "$a" -le "$b" ]

Uslovne strukture:
* If/then           *If/then/else       *if/then/elif/then/else     *CASE
if <USLOV>; then    if <USLOV>; then    if <USLOV>; then            case <VREDNOST (npr $1)> in
    <NAREDBA>       <NAREDBA>           <NAREDBA>                       Prvi slucaj)
    ...             else                elif <USLOV>; then                  <NAREDBE>;;
    <NAREDBA>       <NAREDBA>           <NAREDBA>                       Drugi slucaj)
fi                  fi                  else                                <NAREDBE>;;
                                        <NAREDBA>                           * (svi ostali slucajevi))
                                        fi                                  <NAREDBE>;;
                                                                    esac

* Petlja while - radi dok je uslov ispunjen.          *Petlja until: Izvrsava se sve dok [ USLOV ] nije ispunjen.
while [ USLOV ] npr. [ $i -ne 4 ]                     until [ USLOV ]
do                                                    do
    <NAREDBA>                                             <NAREDBA>
    ...                                                   ...
    <NAREDBA>                                             <NAREDBA>
done                                                  done

Funkcija: Deo koda unutar nje se ne izvrsava dok se ne pozove, obicno se poziva na kraju skripte.
function <IME FUNKCIJE> {
    <NAREDBA>
    ...
    <NAREDBA>
}
<IME FUNKCIJE> (<PARAMETAR 1> ... <PARAMETAR N>, paramteri ako unutar naredbi ima nesto tipa echo $1)

Vezba sa casa:
1. Dodeliti vrednost varijabli
2. Ispisati vrednost varijable
3. Ispisati sabranu vrednost varijable i sabranu sa 15.

#!/bin/bash
i=55
echo "i=$i"
echo $[$i+15]
chmod +x <naziv skripte>
./<naziv>
```

Zadatak: Skripta mora da procesira 3 argumenta: Napraviti fajl u prvom folderu. Svi mogu da pokrenu.
$1 - Ime datoteke $2 - Apsolutna putanja do prvog foldera $3 - -ll- do nekog drugog foldera

```
#!/bin/bash
mkdir $2
mkdir $3                    chmod 777 script
touch $2/$1                 ./script Fajl PrviFolder DrugiFolder
```

AIOS2 05 - Pristup na udaljene sisteme pomoću OpenSSH servisa

- OpenSSH se koristi za bezbedno pokretanje shell programa na udaljenom računaru. Potrebno je da korisnik ima nalog na udaljenom sistemu.
- Pokreće se naredbom **ssh**, nalog ima isto ime na oba sistema:
* **ssh <ime ili IP adresa udaljenog racunara>**
* ssh alpha (trenutni korisnik) **ili** ssh student@alpha (nalog student).

- Komanda **w** prikazuje listu svih prijavljenih korisnika. Dodatkom argumenta -f vidimo polje from koje nam govori odakle se povezuje korisnik (:0 za lokalni računar ili :neki_IP za udaljeni).

User	TTY	From	LOGIN@	IDLE	JCPU	PCPU	WHAT
lalas	:0	:0	Wed10	0.00s	1:17	0.14s	gdm-session-work

- Kad se ssh klijent povezuje na server, pre prijavljivanja korisnika server klijentu šalje kopiju svog javnog ključa. Javni ključ servera korisnik snima u ~/.ssh/known_hosts.
- Ako dođe do promene javnog ključa servera, neće moći da se obavlja komunikacija sve dok klijent ne ažurira svoju kopiju javnog ključa servera.
- To se radi brisanjem ~/.ssh/known_hosts datoteke, nakon čega će kod prvog narednog uspostavljanja konekcije server poslati klijentu novu kopiju svog javnog ključa.

PKI autentifikacija
- Kod SSH pristupa moguća je i autentifikacija bez lozinke sa sistemom baziranom na paru PKI ključeva.
- Dolazi do kreiranja para ključeva za korisnika, **javnog** i **privatnog** koji se prave na PC-u kom se pristupa.
- Privatni je obezbeđen dok se javni kopira na računare koji pristupaju, oni su matematički povezani.
- Usled korišćenja RSA algoritma, teško je saznati privatni na osnovu javnog. **ssh-keygen** generiše par klj.
- Lokacija privatnog ključa je ~/.ssh/id_rsa, javni je ~/.ssh/id_rsa.pub
- Pre korišćenja javni ključ mora biti prekopiran na udaljen računar. **ssh-copy-id [username]@<ime/IP>.**
- SSH passphrase omogućava zahtev lozinke pri pristupu javnom ključu.
- Da ne bismo kucali lozinku svaki put: ssh-agent bash, ssh-add, unos lozinke . Važi samo za trenutnu ses.

- Konfiguraciona datoteka se nalazi u **/etc/ssh/sshd_config**.
- Kucamo vim /etc/ssh/sshd_config, i onda menjamo pretragom npr. /PermitRoot pa u Insert mode.
- Blokiranje spoljnog pristupa na nalog root:
* Sa **#PermitRootLogin** yes na #PermitRootLogin no
- Zabrana pristupa korišćenjem lozinke, ograničavanje pristupa na metod para PKI ključeva.
* Sa **#PasswordAuthentication** yes u #PasswordAuthentication no
- Da bi promena bila efektivna, treba restartovati ssh demon : **systemctl restart sshd.**

AIOS2 06 - Konfigurisanje planiranih zadataka

- Definišemo pokretanje skupa naredbi u određenom trenutku u budućnosti.
- Jedan od načina konfigurisanja planiranih zadataka je preko sistemskog demona **atd**. interakcija sa demonom se obavlja preko naredbi **at** i **atq**.
- Demon atd obezbeđuje do 26 redova za izvršavanje planiranih zadataka a-z sa opadajućim prioritetom od reda a ka redu z.

- Sintaksa: **at <TIMESPEC> < <SCRIPT>**
** at now +5min < backupscript*
- Za pokretanje pojedinačnih komandi: **<COMMAND> | at <TIMESPEC>**
** echo "Hello" > echo.log | at now +1min*

- Opcijom **-q <QUEUENUMBER>** menja se red za čekanje. Podrazumevani red za čekanje je a red.
** at -q b 14:00 < /scripts/scr.bash*
job 14 at Wed Mar 23 14:00:00 2016

Postoje razni načini za pisanje **<TIMESPEC>:**
* now +5min ; teatime tomorrow (16h) ; noon + 4days (U podne za 4 dana) ; 5pm august 3 2016 ; 18:00 +3days => * *echo hello > hello.log | at 18:00 march 24 2016*

- Pregled planiranih zadataka koji čekaju na izvršvanje: **atq** ili **at -l.**
*[root@localhost Desktop]#at -l **ili** atq*
8 Thu mar 24 18:00:00 2016 a root
<JOBNUMBER> <DATE> <QUEUENUMBER> <VLASNIK (u čijem kontekstu se pokreće)**>**

- Pregled komande koja se izvršava kao deo planiranog zadatka se gleda sa **at -c <TASKNUMBER>**
** at -c 8*

- Uklanjanje planiranog zadatka se radi preko **atrm <JOBNUMBER>**
** atrm 8*

CRONTAB
- Korisnici bez admin privilegija koriste komandu **crontab** za upravljanje planiranim zadacima.
crontab -l -r -e: Lista/uklanja/kreira ili modifikuje sve planirane zadatke aktuelnog korisnika.
crontab <ime datoteke> : Uklanja sve planirane zadatke i zamenjuje ih zadacima pročitanim iz datoteke.
- Crontab -e modifikuje konf. datoteku **/var/spool/cron/<username>** . Datoteka ne sme direktno da se menja. Crontab se ne koristi za upravljanje sistemskih zadataka.
- root može da koristi opciju **-u <username>** za upravljanje zadacima drugih korisnika.
- Pokretanjem naredbe **crontab -e** startuje se vi editor. Jedan planiran zadatak zauzima jednu liniju teksta, prazne linije teksta su dozvoljene, a komentari počinju tarabom #.
- Varijable okruženja se koriste u formatu NAME=value, uobičajne varijable su **SHELL** i **MAILTO**.
- SHELL menja shell aplikaciju koja izvršava komande, MAILTO mail na koji se šalje izlaz.

- Svaki planirani zadatak se sastoji od 6 polja koji određuju kada i šta će biti pokrenuto:
MINUT* *SAT* *DAN U MESECU* *MESEC* *DAN U SEDMICI* *NAREDBA
- Ukoliko su definisana oba polja dan u mesecu i dan u sedmici onda će se u oba vremena pokrenuti.

Prvih pet polja imaju ista sintaksa pravila:
- Karakter * znači uvek
- Dani u nedelji se reprezentuju brojem 0 = nedelja, 1 ponedeljak ... 7 = nedelja
- Opseg se definiše kao x-y i ukljucuje x i y.
- Lista termina se definiše kao x,y.
- Lista u koloni minut može da uključuje i opsege npr. 5,10-13,7
- U koloni minut izraz */x indicira interval od x npr. */7 znači da zadatak započinje na svakih 7 minuta.
- U koloni mesec možemo koristiti skraćenice Jan, Feb...

1. crontab -e
* Otvori se VI editor.

```
#minut      sat danumesecu mesec danusedmici komanda
   0        16      *       *       *        ....
   5        9-16    *       *       *        ....
5,6-7,10    9-16    *       *       *        ....
*/10        9-16    *       *       *        ....
*/10        9-16    *       *     Mon,Tue    ....
  15         9      1       Apr     *        ....
```

```
0 9 2 2 * /usr/local/bin/yearly_backup
Izvrsava komandu /usr/local/bin/yearly_backup u 9h drugog februara svake godine

*/7 9-16 * Jul 5 echo "Chime"
Salje e-mail poruku koja sadrzi rec Chime na svakih 7 minuta u periodu 9-16h svakog petka u julu.

58 23 * * 1-5 /usr/local/bin/daily_report
Izvrsava komandu /usr/local/bin/daily_report u 23h 58min svakog radnog dana.
```

Sistemski planirani zadaci
- Oni se konfiguršu direktnom modifikacijom konfiguracionih datoteka.
- Primarna konfiguraciona datoteka sistemskih planiranih zadataka je **/etc/crontab** ili **/etc/cron.d** .
- Postoje i folderi **/etc/cron.hourly | daily | weekly | monthly**, u tim folderima se izvršavaju sve skripte koje se nalaze svakog sata, dana, nedelje, meseca.
- Razlika između korisničkih i sistemskih planiranih zadataka je dodatno polje koje definiše nalog .
MINUT* *SAT* *DAN U MESECU* *MESEC* *DAN U SEDMICI* *KORISNIK* *NAREDBA
- Dnevni, sedmični i mesečni zadaci se izvršavaju kroz **/etc/anacrontab**.
- Svi koriste komandu run-parts za izvršavanje skripti.

```
*Koliko cesto se zadatak izvrsava (u danima)    *Koliko se ceka pre izvrsenja zadataka  *Naziv datoteke iz /var/spool/anacron   *Naredba koja se pokrece
1   5   cron.daily      nice run-parts /etc/cron.daily
7   25  cron.weekly     nice run-parts /etc/cron.weekly
@monthly 45 cron.monthly        nice run-parts /etc/cron.monthly

- Anacron se koristi za sigurno pokretanje servisa bez obzira da li smo restartovali racunar ili ne.
```

AIOS2 07 - Dodavanje diskova particija i sistema datoteka

- Master Boot Record omogućava podizanje OS-a na računarima koji koriste **BIOS**, poseduje tabelu particija (sa 4 polja za opis particija), i deo sa programom koji omogućava podizanje računara u ranoj fazi.
- **GPT** diže OS na **UEFI** računarima, poseduje samo tabelu particija (128 particija) i ima kopiju na početku i kraja diska radi redundantnosti.
- MBR diskovi koriste MBR strukturu, imaju do 4 primarne particije (može 1 extended), veličina do 2TB.
- GPT diskovi koriste GPT strukturu podataka, do 128 particija, veličina do 8 ZiB (zebibajt - milijardu TiB).

Podizanje sistema koji koristi BIOS
1. BIOS POST
2. Boot sequence (zavisi od konfiguracije BIOS-a) (koji uređaj se prvi pokreće (HDD, CD-DVD...))
3. Učitavanje boot sector-a uređaja (prvih 512 bajta podataka) - MBR
Kod Windows-a u tabeli traži particiju markiranu kao aktivna i startuje njen boot sector (prvi sektor) u kome se nalazi program koji dalje učitava OS.; dok kod Linux postoji specifičan boot loader tipa GRUB koji radi isto.

Podizanje sistema koji koristi UEFI
1. BIOS POST
2. Boot sequence
3. Učitavanje boot programa na specijalnoj particiji tzv. EFI sistemska particija koji dalje učitava OS.

- fdisk radi samo sa MBR, gdisk radi sa oba. *f/gdisk /dev/<ime diska ili particije>*

```
- Desni klik na CentOS VM1 > Settings > Storage > SATA > Add Hard Disk > Next... Napravimo 4 hard diska (VD1, VD2, VD3 i VD4).
1. ls -la /dev | grep sd => Listanje svih diskova i particija
- sda i sdb koristimo kao MBR diskove
2. fdisk /dev/sdb - Program ne snima promene dok ih ne upisemo. Slovo m za listanje naredbi.
- p = Prikaz tabele particija
- d = Brisanje particija
2.1 n = New partition => Potom biramo da li je primarna (maksimum 4) ili extended particija.
2.2 Koji broj particije => Samo enter (default)
2.3 Lokacija prvog sektora => Samo enter (default)
2.4 Lokacija poslednjeg sektora => +K - kilobajt +M - megabajt +G - gigabajt .  +1G = Particija je velicine 1G.
3. Pisemo W, cuvaju se promene i izlazi se iz programa.
- Potom opet kucamo fdisk /dev/sdb i p da vidimo tabelu particija, gde vidimo nasu novu particiju.
- ID kod particije se moze saznati naredbom l, gde vidimo listu podrzanih tipova particija.
- ID ove particije je 83 => Linux.
- Napravimo 4 particije
4. d, potom izaberemo particju broj 4 i ona je obrisana.

1. t - Izmena ID-a particije
2. Kucamo broj particije
3. Kucamo Hex code tj. nov ID
4. Kucamo p da vidimo rezultat

- Za GPT particije koristimo program gdisk /dev/sdd , ? za listu naredbi. Slican prethodnom programu.
```

- Kreiranje sistema datoteka (formatiranje particije) : **mkfs -t <tip> <putanja do particije>**
- Manuelno de/montiranje : **u/mount <putanja> <folder za montiranje>.**
- Nakon restarta neće doći do ponovnog montiranja, za to se koristi datoteka /etc/fstab
<UUID / PUT DO PARTICIJE> <MOUNT FOLDER> <TIP> DEFAULTS <DUMP> <FSCK>
- Dump (0/1) određuje da li dolazi do kreiranja rezervne kopije, fsck redosled provere particije na greške tokom dizanja OS-a pomoću fsck posle nekoretnog demontiranja particije; 0 znači da ne proverava.

Swap prostor
- Prostor diska koji se koristi kao dodatak radnoj memoriji za smeštanje neaktivnih stranica.
- Zbir RAM-a (tj. operativne memorije) i swap prostora naziva se virtuelna memorija.
- Kada potrošnja pređe limit RAM-a, kernel kopa po njemu i traži neaktivne stranice koje upisuje u swap prostor a njenu adresu dodeljuje drugom procesu. Po potrebi, one se prebacuju nazad u RAM.

```
1. Kreiranje particije.
2. Podesavanje tipa particije na 82 - Linux Swap za MBR disk ili 8200 za GPT disk.
3. Kreiranje swap potpisa particije : mkswap /dev/sdb3
4. Aktivacija swap prostora:
* Neperzistentna: swapon /dev/sdb3
* Perzistentna: Particija mora biti konfigurisana u fstab datoteci.
    * Primer: UUID=fbd7fa60-b781-44a8-96b1-37ac3ef534bf swap swap defaults 0 0
    * Komanda swapon -a aktivira sve swap prostore definisane u datoteci fstab.

- Deaktivacija swap prostora: swapoff /dev/sdb3
- Pregled svih aktuelnih prioriteta komandom swapon -s .
```

- **du -ch /home** (za foldere); **df -h /dev/sda** (za particije); **fsck /dev/sdc** (zahteva demontiranje prvo)

NAREDBE:
01. ps (lista procesa), jobs (pozadinski), fg /bg%<JID>, kill <PID>, killall <IME_PR>, w, pkill, pgrep, uptime.
02. logrotate, logger, tail -f /var/log/messages ili journalctl, timedatectl, tzselect, chronyc sources -v.
03. #!/bin/bash -x, #, \ , ‘’, $, for, while, until, function, pwd, ++/--<VARIJABLA>, <VARIJABLA>++/-- .
04. script $
/*/@/#/? exit, echo $?, [<vr.> <operator> <vr.>], &&, ||, if/then/elif/else, case.
05. ssh [user]@<ime/IP udaljenog>, w -f, ssh-keygen, ssh-copy-id, ssh-agent-bash, ssh-add.
06. at -c/q/l, atq, atrm, crontab -l/e/r, run-parts.
07. fdisk, gdisk, mkfs, mount, umount, blkid, man, mkswap, swapon, swapoff, du, df, fsck.

----- KRAJ PRVOG DELA -----

AIOS2 08 - Apstrakcija disk prostora korišćenjem LVM-a

- LVM omogućava grupisanje više fizičkih diskova u jedno skladište.
- Fizički uređaji (Physical devices): Diskovi, particije, RAID sistemi, SAN diskovi.
- Fizički volumen (Physical Volume - PV): Koristi se za registrovanje fizičkog uređaja kao dela grupe.
- Grupa volumena (Volume Group - VG): Kolekcija PV. Jedan PV se može dodeliti samo jednoj grupi.
- Logički volumen (Logical Volume - LV): Kreira se na prostoru grupe volumena i predstavlja LVM ekvivalent particijama. 1. fdisk 2. pvcreate 3. vgcreate 4. lvcreate
- Logički volumen se kreira iz 4 koraka: Kreiranje particija, kreiranje fizičkog volumena na njima, kreiranje grupe volumena na fizičkim volumenima i kreiranje LV-a na grupi volumena.
- Uvećanje znači dodavanje fizičkih volumena grupi, smanjenje znači njihovo uklanjanje iz VG.
- Tanki logički volumen može imati proizvoljnu veličinu nezavisnu od realnog slobodnog prostora na disku.

AIOS2 08 - Apstrakcija disk prostora korišćenjem LVM-a

Kreiranje fiksnog logičkog volumena i njegovo montiranje

1. fdisk => N => Enter =>Enter => +256M ; t => Enter=> 8e (LVM); w

Dobijemo sda1, sda2, sda3.

2. partprobe
3. pvcreate /dev/sda[1-3]
4. vgcreate shazam /dev/sda[1-3]
5. lvcreate –L 750M –n lv-alpha shazam
6. mkfs –t xfs /dev/shazam/lv-alpha
7. mkdir /mnt/lv
8. vim /etc/fstab

/dev/shazam/lv-alpha /mnt/lv xfs defaults 0 0

9. mount –a
10. lvdisplay /dev/shazam/lv-alpha
11. df –h

Kreiranje tankog logičkog volumena

4. lvcreate –L 750M –T shazam/tp1
5. lvcreate –V800M –T shazam/tp1 –n ThinPool

Kreiranje RAID-a

1. lvcreate --type raid1 –m 1 –L 3G –n Raid1 shazam
2. lvcreate --type raid5 –i 3 –L 3G –n Raid5 shazam

Proširivanje grupe volumena i logičkog volumena

1. fdisk => …. Dobijemo sda4
2. partprobe
3. pvcreate /dev/sda4
4. vgextend shazam /dev/sda4
5. lvextend –L 800M /dev/shazam/lv-alpha
6. xfs_growfs /mnt/lv **ILI** resize_2fs /dev/shazam/lv-alpha
7. df –h /lv-alpha , vgdisplay, lvdisplay…

Uklanjanje particije iz grupe volumena

1. pvmove /dev/sda4
2. vgreduce shazam /dev/sda4

Uklanjanje logičkog volumena

1. umount /mnt/lv
2. lvremove /dev/shazam/lv-alpha
3. vgremove shazam
4. pvremove /dev/sda[1-3]

AIOS2 09 – Korišćenje softverskog RAID-a

RAID 0 : Striping bez pariteta, min. broj diskova 2. Najbolje poboljšava performanse i kapacitet, ali ako jedan crkne svi odu. **Superblock** - struktura podataka koja sadrži informacije o Linux softverskom RAIDu.
RAID 1 : Mirroring sistem koji obezbeđuje bezbednost sistema, zasniva se na 2 hard diska gde se podaci upisani u jedan disk automatski zapisuju i u drugi pomoću kontrolera. ***RAID 10** = Basic nivo R1, sek. R0
RAID 3 : Uvodi paritet, min. br. diskova je 3 (1 za paritet). Ako jedan disk crkne, koristeći XOR operaciju na preostala dva (sabiranje bez prenosa nule) možemo da povratimo stare podatke na novi disk.
RAID 4: Sličan RAID 3 ali bolji u nekim situacijama. MBD je 3, 1 za paritet, dimenzije traka su veće od R3.
RAID 5 : Podaci o paritetu se nalaze na svim diskovima (dijagonala), rešava disk hammering problem.
RAID 6 : Podaci o paritetu se snimaju na još jedan disk (MBD je 4), sistem radi čak i da izgubimo 2 diska.
MDF RAID : Sličan RAID-u 6, otporan na pad više od 2 diska, zahteva još 1 disk (3 diska) za paritet.

Priprema uređaja za implementaciju
1. mdadm --examine /dev/sd[b-c][1-2]

Kreiranje RAID1
1. mdadm --examine /dev/sdc[1-2]
2. mdadm --create /dev/raid1 --level=1 --raid-devices=2 /dev/sdc[1-2]
3. mdadm --examine /dev/sdc[1-2] **ILI** mdadm --detail /dev/raid1 **ILI** ls -la /dev/raid1
4. mkfs -t xfs /dev/raid1
5. mkdir /mnt/raid1
6. vim /etc/fstab
/dev/raid1 /mnt/raid1 xfs defaults 0 0
7. mount -a

Dodavanje spare diska i multimirror-a => Omogućava oporavak sistema u slučaju pada 1 diska
1. mdadm --manage /dev/raid1 --add /dev/sdc3
2. mdadm --grow --raid-devices=3 /dev/raid1

Uklanjanje RAID-a
1. umount /dev/raid1
2.1 mdadm --zero-superblock /dev/sdb[1-2]
2.2 mdadm --stop /dev/raid1

Kreiranje RAID5
2. mdadm --create /dev/raid5 --level5 --raid-devices=3 /dev/sdc[1-3]

Dodavanje diska RAID-u
1. mdadm --examine /dev/sdc4
2. mdadm --manage /dev/raid5 --add /dev/sdc4
3. mdadm --detail /dev/raid5

AIOS2 10 - Pristup sistemima datoteka i uređajima

- Sistem datoteka - struktura podataka za smeštaj fajlova i foldera, može biti na disku ili particiji.
- Njena hijerarhija pristupa sis. d-teka preko stabla direktorijuma, sa jednim korenskim / direktorijumom.
- Dodavanje novih SD u postojeće stablo naziva se montiranje, a folder gde se SD montira je mnt. point.
- Svi uređaji namenjeni skladištenju podataka nazivaju se **blok uređaji**. LVM njih spaja u grupu volumena.

Alati za analizu preostalog disk prostora

- **df** => Prikazuje podatke za sve particije .
* df -t xfs => Samo za xfs fajl sisteme.
* df -H /dev/sda1 => Samo za sda1 particiju (malo h za SI prefikse).
- **du** => Prikazuje količinu prostora koje zauzimaju svi podirektorijumi trenutnog foldera.
* du -aH

Montiranje i demontiranje sistema datoteka

- **mount** /dev/vdb1 /mnt/mydata
* mount **UUID**="<broj>" /mnt/mydata
- **blkid** => Lista diskova i particija sa njihovim UUID-evima.
- **umount** /mnt/mydata - *Demontiranje nije moguće ako procesi pristupaju sadržaju, zato prvo killall.*
- **lsof** | grep sdc1 => Lista kojim resursima pristupaju aktivni procesi.
- **kill** ili **killall** => Ubijanje procesa.

Kreiranje linkova između datoteka

- **ln (-s) <target> <link name>**
* ln /etc/hosts hosts => Sada npr. cat hosts je isto kao i cat /etc/hosts
- Ako obrišemo sve hard linkove prostor koji taj fajl zauzima se oslobađa, dok soft linkovi još uvek pokazuju na datoteku iako je ona obrisana (Dangling Soft Link). Soft l. mogu da pokazuju ka datotekama.

Lociranje datoteka - *find <lokacija> [uslov (name/perm/size/user/group/uid/gid)] [ime]*

- **updatedb** => Manuelno ažuriranje baze podataka.
- **locate etc** => Traži sve datoteke i direktorijume koji u putanji imaju izraz etc.
* locate -b '\etc' => Traži sve datoteke koje se tačno zovu etc.
* locate -i messages => Koji u putanji imaju izraz messages ali neosetljivo na mala i velika slova u nazivu
* locate -n 6 messages => Ograničava broj rezultata na 6.

AIOS2 11 - Kopiranje podataka između sistema

- **scp /etc/yum.conf /etc/hosts sfiles:/home/student** => Kopiranje sa našeg na udaljen računar.
- **scp web1:/etc/hosts /home/pp** => Kopiranje hosts sa računara web1 u naš /home/pp folder.
- **scp -r root@sfiles:/var/log /tmp** => Rekruzivno kopiranje kao root sa udaljenog u naš /tmp folder.

- **sftp serverx** => Potom nas pita za login, pristup udaljenom računaru serverx. Rade naredbe tipa ls, cd ...
* **exit** - izlaz iz sesije
* **put** /etc/shots - Upload naše datoteke u trenutni direktorijum na udaljem sistemu. Radi samo sa files.
* **get** /etc/yum.conf - Download datoteke sa udaljenog računara u trenutni folder.

- **rsync -av /var/log/ /tmp** => Sinhronizacija 2 lokalna foldera. -n za simulaciju sinhronizacije.
- **rsync -av /var/log/ sfiles:/tmp** => Sinhronizuje lokalni /var/log sa udaljenim (sflies) /tmp.
- **rsync -av desktops:/var/log /tmp** => Sinhronizuje udaljeni /var/log sa lokalnim /tmp.

AIOS2 12 - Konfigurisanje komunikacione barijere

- **Firewall:** Softver koji omogućava filtriranje paketa mrežnog saobracaja (**Packet Filtering**), kao i filtriranje sa uspostavom stanja (**Stateful Filtering**).
- Paketno filtriranje se zasniva na analizi zaglavlja protokola transportnog i internet sloja TCP/IP modela.
- Elementi paketnog filtera: IP adresa izvora i odredišta, TCP/UDP port izvora i odredišta i broj protokola.
- Firewall briga za sadržaj paketa, on samo pregleda ovo iznad spomenuto.
- Stateful filtering sprečava napade na transportnom sloju praćenjem uspostavljenih veza i onemogućava napadača da se umetne u postojeću komunikaciju.
- **Aplikaciono filtiranje** na bazi sadržaja paketa odlučuje da li on može proći ili ne. (npr. kod HTTP paketa može biti pojava ključnih reči, npr. ne želimo da se prikazuju stranice koje sadrže nasilje i sl.)
- Ap. filtriranje je npr. propuštanje samo FB-a moblilnim telefonima u onim akcijama gde je FB besplatan.
- Stateful čita tabelu stanja i upoređuje sa ip adresom, paketni filter čita samo zaglavlje, dok aplikaciono zauzima najviše CPU resursa jer mora da čita celi paket.
- **Host-Level Firewall** štiti samo računar na kom je pokrenut, **Network-Level Firewall** štite čitavu mrežu.
- Podrazumevani softver je **firewalld**. On upravlja podsistemom Linux jezgra koji se zove netfilter.

***** Ugasimo Centos-VM2, Settings > Network > na Adapter 2 dodamo NAT => Sada mašina može na net.

- Firewall raspodeljuje dolazni saobraćaj na **zone**, pri čemu svaka zona ima svoj skup pravila komunikacije. Zona definiše nivo poverenja prema dolaznom saobraćaju.

Firewalld koristi sledecu logiku:

1. Ako se izvorna IP adresa dolaznog paketa poklapa sa konf. pravilom zone, paket se usmerava toj zoni.
2. Ako se dolazni mrežni interfejs poklapa sa konf. interfejsa zone, paket se usmerava toj zoni.
3. U suprotnom, koristi se default zona (public). Postoji više predefinisanih zona (dmz, work, public...).

- Pošto je **firewalld** u konfliktu sa njima, prvo systemctl mask ip/e/tables/6.service
- Zone: trusted, home, internal, work, public, external, dmz, block, drop
- **firewall-cmd** ili **firewall-config** za podešavanje, konf. datoteka je u /etc/firewalld.
- **--permanent** za trajnu promenu, ukoliko nije prisutna onda je samo runtime (trenutna) konf.
- Posle konfigurisanja **--reload**, za specifičnu zonu **--zone=<naziv>** (default zona je public).
- Ako se izostavi zona argument, sve promene se odnose na default zonu (tj. public).
- Pomoću **--timeout=<vreme u sekundama>** možemo da ograničimo vreme promene.
- --**get/set-default-zone** => Daje informaciju ili podešava default zonu.
- **--get-zones/services/active-zones** => Lista dostupne zone/predefinisane servise/aktivne zone.
- **--add/remove-source=<CIDR> [--zone=<zona>]** => Usmerava/uklanja saobraćaj sa CIDR u tu zonu.

- <CIDR> - je IP Network ID mreže sa maskom npr. 172.168.16.0/16

- **--add/change-interface=<Interfejs> [--zone=<zona>]** => Sa novog/postojećeg interfejsa na int. u zonu.
- **--list-all [--zone=<zona>]** ili **--list-all-zones** => Lista sve podatke sa određene ili sve zone.
- **--add/remove-service=<servis> [--zone=<zona>]** => Dodaje/uklanja podršku za servis.

- --add/remove-port=<port/protokol> => Dodaje/uklanja port proizvoljnog servisa.

- **firewall-cmd --get-services** => Lista moguće servise za instalaciju.

1. firewall-cmd --permanent --add-service=mysql --add-port=5766/TCP

Rich rules

- **rule** [family=”ipv4|ipv6”], ako nema onda se primenjuje na oba.
- **source/destination address**=”address[/mask]” [invert=”True”] - Inv. se odnosi na sve osim date adr.
- **service name**=”service name” - Filtriranje paketa koji sadrže protokol odabranog --get-services servisa
- **port port**=”port value” **protocol**=”tcp|udp”
- **protocol value**=”protocol value”
- **icmp-block name=**”icmptype name” - Lista icmp paketa se dobija preko **firewall-cmd --get-icmptypes**.
- **masquerade** - maskiranje
- **forward-port port=**”port value” **protocol=**”tcp|udp” **to-port**=”port value” **to-addr**=”address”

* **Prosleđivanje porta** tj. zamena porta izvorišne ili odredišne aplikacije.

- **log [prefix=**”text”**] [level**=”log level (0-7)”**] [limit value**=”rate/duration (1/m)”**]**
- **accept** | **reject** [type=”reject type”] | **drop =** propušta/odbija sa porukom/odbija bez slanja odgovora.
- Pravila se procesiraju: port forwarding, masquerading rules > logging > allow > deny rules
- **--add/remove/query-rich-rule=’<RULE>’ [--zone=<zona>] -** Pravila se vide i u --list-all

*** firewall-cmd --permanent --zone=dmz --add-rich-rule=’rule family=ipv4 source address=172.16.0.10/32 reject’** => success.

*** firewall-cmd --add-rich-rule=’rule service name=ftp accept’**

*** firewall-cmd --add-rich-rule=’rule protocol value=esp drop’**

- **NAT** tehnologija vrši translaciju privatnih IP adresa u javne i obrnuto, kao i translaciju portova. Kod firewalld translacija IP adresa naziva se **maskiranje**, a translacija portova **port forwarding.**

Maskiranje (translacija adresa) pomoću osnovnog i bogatog pravila

1. firewall-cmd --permanent --add-masquerade
1. -||- [--zone=dmz] --add-rich-rule=’rule family=ipv4 source address=192.168.0.0/24 masquerade’

Prosleđivanje porta pomoću osnovnog i bogatog pravila

1. firewall-cmd --permanent --add-forward-port=port=513:proto=tcp:toport=132:toaddr=192.168.0.254

* Dolazne veze na port TCP 513 barijere ka zoni public prosleđuje na TCP port 132 na PC sa adresom 192.168.0.254

1. -||- --add-rich-rule=’rule family=ipv4 source address=192.168.0.0/26 forward-port port=80 protocol=tcp to port-8080’ => Dolazne veze na port TCP 80 iz ^ prosleđuje na TCP 8080 same barijere

Vežba:

Serverx:

1. firewall-cmd --permanent --add-rich-rule=' rule family=ipv4 source address=172.16.0.10/32 forward-port port=443 protocol=tcp to port=22'
2. firewall-cmd --reload

Desktopx:

1. ssh -p 443 serverx

AIOS2 13 - Rešavanje problema u procesu podizanja operativnog sistema

- PC firmver je prvi softver koji se inicijalizuje nakon paljenja računara (nama je poznat kao BIOS).
- To je softver koji vrši proveru i pokretanje svih hardverskih komponenti računara (POST).
- UEFI se koristi kod modernih računara. Bolja bezbednost, brže podizanje sistema posle hibernacija, sposobnost podizanja sistema sa diskom vecim od 2TB...
- Boot loader je program koji biva pokrenut od strane firmvera, učitava se sa hard diska u operativnu memoriju i pokreće kernel OS-a time predajući kernelu proces podizanja istog.

Proces podizanja na BIOS sistemima

1. Paljenje računara, startuje se POST firmvera.
2. Firmver traži uređaj konfigurisan da se sistem preko njega diže i pokreće GRUB loader faze 1 iz MBRa diska.
3. Grub 1.5 se učitava, smešten između MBRa i prve particije, sposoban za čitanje particija, startuje Grub 2.
4. Grub boot loader faze 2 se učitava, on je smešten na prvu particiju (sda1) u folder /boot/grub2 .
5. Boot loader učitava konf. sa diska i prikazuje nam meni sa dostupnim konf. OS-a (podaci iz ^/grub.cfg)
6. Nakon selekcije OS-a, loader učitava **kernel** (**vmlinuz**) i **initframs** (initial RAM file system) sa diska i smešta ih u memoriju. Initframs sadrži module za uređaje neophodne pri dizanju OS-a, init skripte i dr.
7. BLoader predaje kontrolu nad sistemom kernelu prosleđujući mu opcije i lokaciju **initframs**-a u mem.
8. Kernel pokreće hardver čije drajvere može naći u initframsu i onda izvršava /sbin/init => **systemd .**
9. Systemd sa initframsa izvršava sve systemd jedinice iz initrd.target uz montiranje korena sistema dat.
10. Koren sis. datoteka se menja iz korena initrfarms sis. dat. na realni sistem datoteka na disku.
11. Systemd traži prekonfigurisani target (/etc/systemd/system/default.target) i onda startuje jedinice.

Proces podizanja na UEFI sistemima

3. UEFI firmver pokreće UEFI boot loader, prva particija UEFI sistema je EFI sistemska particija montirana na /boot/efi. U folderu /boot/efi/EFI/BOOT nalazi se EFI boot loader **BOOTX64.EFI**
4. Na CENTOS sistemima **BOOTX64.EFI** učitava GRUB2 iz /boot/efi/EFI/centos

Podizanje BIOS sistema

1. ls -la /dev | grep sd
2. fdisk /dev/sda => p
3. df -h [putanja do particije], pvs, vgs, lvs
4. cd /boot
5. ls -la => Vidimo grub i grub2, kao i razne verzije initframs-a
6. cd grub2 ; ls -la => devices, grub.cfg ...
7. less grub.cfg => Tražimo /menuentry
8. cat grub.cfg | grep "menuentry 'C" => Vidimo dve linije koje imamo kada podižemo OS (menuentry 'CentOS Linux, with Linux 3.10.0...)

Podizanje UEFI sistema

4. cd /boot/efi/EFI/BOOT
5. ls -la => Vidimo BOOTX64.EFI i fallback.efi
6. cd ..
7. cd centos ; ls -la
8. less grub.cfg - /menuentry i nadjemo tamo gde pise CentOS.

Ispravka greški

1. su ; vim /etc/fstab => Umesto foldera /home za montiranje izmenimo u /hom
2. reboot
3. Stigne do login ekrana, ali kad krenemo da se prijavimo ništa se ne dešava, to je zato što je disk montirao na /hom a tamo nema podataka o korisnicima.

Ovaj problem se rešava privremenom promenom menuentry bloka.

- Palimo sistem, biramo konfiguraciju, pritisnemo e, lociramo liniju početka konfiguracije kernela:
- String linux na 64 Bit IBM Power Series, linux16 na x86-64 BIOS i linuxefi na x86-64 UEFI sistemima.
- Pomerimo kursor na kraj linije (taster end) i editujemo ili dodajemo željeni parametar. Ove izmene se odnose samo na trenutni proces podizanja sistema.

4. Kod stringa linux16, na kraju upisujemo init=/bin/sh
- Otvori nam terminal
5. mount -o remount,rw / => Ponovo montira, bez ovoga neće da upiše ništa u /etc/fstab (read only je)
6. vi /etc/fstab => Promenimo /hom u /home => Sad kada restartujemo radi.

Resetovanje konfiguracije i reinstalacija (ako ima korumpiranih datoteka)

1. rm /etc/grub.d/* i rm /etc/sysconfig/grub
2. grub2-install /dev/sda (BIOS) **ILI** yum reinstall grub2-efi shim (UEFI)
3. grub2-mkconfig -o /boot/grub2/grub.cfg (BIOS) ILI grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg

AIOS2 14 - Pristup preko Server Message Block - SMB protokola

SMB je protokol namenjen mrežnom pristupu na datoteke, printere...

Serverx:

Korišćenje samba servera kao file servera

1. yum install samba
2. systemctl enable smb
3. systemctl start smb
4. systemctl status smb
5. firewall-cmd --permanent [--zone=public] --add-service=samba
6. firewall-cmd --reload
7. mkdir /sambashare
8. touch /sambashare/file1
9. useradd petar ; passwd petar

Kreiranje SAMBA naloga za korisnika i podesiti lozinku

1. pdbedit -a -u petar
2. pdbedit -L => Lista samba naloge (petar:1001:)

Konfigurisanje konfiguracione datoteke
1. vi /etc/samba/smb.conf , u segmentu Share Definitions:
[sambashare] (Sami dodamo)
path = /sambashare
public = yes
writable = yes
valid users = petar
2. testparm => Provera ispravnosti konf. datoteke
3. systemctl reload smb
4. yum install samba-client => Instalacija samba klijenta
5. smbclient -L localhost -U petar => Lista deljenih foldera sa korisnikom petar. (traži šifru)

Podesiti SE linux parametere
1. getenforce => Enforcing
2. getsebool -a | grep samba => True/False parameteri (ispisuje da li su on ili off)
3. setsebool -P samba_export_all_rw on (Niti je imalo exporta niti čitanja/upisivanja)

Pregled samba log datoteke i sistemske log datoteke

1. cat /var/log/samba/log.smbd

2. tail /var/log/messages

--- KRAJ PODEŠAVANJA SERVERA ---

Desktopx:

Pristup sa Samba klijenta

1. yum install samba-client cifs-utils (potreban je i CIFS utils)

2. smbclient -L <ipservera> -U <user> => Povezivanje na server

* smbclient -L serverx -U petar

3. mkdir /smbmount => Lokalni folder za montiranje cifs sistema datoteka

4. mount -t cifs -o user=petar //serverx/sambashare /smbmount

Permanentni pristup
1. vi /etc/fstab
//serverx/sambashare /smbmount cifs username=petar,password=petar 0 0
2. autofs
2.1 yum install samba-client cifs-utils autofs (dodajemo autofs)
2.2 vi /etc/auto.master.d/shares.autofs => Kreirati Master map datoteku.
* /shares /etc/auto.shares
2.3 vi /etc/auto.shares => Kreirati map datoteku
* sambashare -fstype=cifs,credentials=/etc/me.cred ://serverx/sambashare
2.4 vi /etc/me.cred => Kreirati Credentials datoteku
* username = petar ; password = petar ; domain = serverx (opciono)
2.5 chmod 600 /etc/me.cred
2.6 systemctl enable autofs
2.7 systemctl start autofs
2.8 systemctl status autofs
3. Pokušati pristup na shares/sambashare : cd /shares/sambashare