Besturingssystemen opgeloste examenvragen

Voeg user toe zonder gebruik te maken van adduser / addgroup (+-p75)

-toevoegen aan /etc/passwd file

- login:x:userid:groupid:userinfo:homedir:defaultprog
- **login**: naam waarmee de gebruiker kan inloggen op het systeem
- **x**: verwijzing naar geëncrypteerde wachtwoord in /etc/shadow
- **userid**: numerieke en unieke identificatie van de nieuwe gebruiker
- **groupid**: numerieke identificatie van de gebruiker's primaire groep
- **userinfo**: extra informatie over de gebruiker zoals de volledige naam
- **homedir**: de homedirectory (standaard map) van de gebruiker
- **defaultprog**: het programma dat gebruiker krijgt na het inloggen

- wachtwoorden staan in etc/shadow

- user:password:systeminfo
- **user**: login van onze gebruiker zoals hij die kreeg bij de eerste stap
- **password**: enkele spatie invullen (geen tekst want is niet geëncrypteerd)
- **systeminfo**: kan meerdere velden omvatten, eigen aan het systeem (copy paste andere lijn)

-wachtwoorden zijn geëncrypteerd > meer veiligheid

-bestand is meestal ook enkel aan te passen door root (tenzij door passwd command)

-deze wachtwoorden w ook apart opgeslagen zodat users wel het passwd file kunnen zien

-wachtwoord instellen:

-commando: passwd username

-geef je wachtwoord in

-home dir aanmaken in de /home map met als naam de gebruikernaam, commando: mkdir

-kopieer inhoud /etc/skel naar deze directory

-cp /etc/skel/?* /newHomeDir

-aanpassen restricties: chown -R login:group .

-mode aanpassen: chmod 700 .

-restricties nieuwe files:

- .bash_profile extra lijn toevoegen: umask 077 (077 wordt geïnverteerd bij unmask > 700)

-geeft user volledig controle en andere geen controle

-Aan welke voorwaarden moet er zijn voldaan om als gebruiker het passwd commando te kunnen uitvoeren

-de administrator moet toestaan dat je je wachtwoord kan veranderen

-je kan enkel je eigen wachtwoord veranderen, dus niet: passwd anderegebruiker

Leg uit CRON (p128)

-scripts uitvoeren op bepaalde tijdstippen, bv:
    -inhoud /tmp iedere dag verwijderen
    -log-files inkorten
    -backups uitvoeren
-uitvoeren administratieve taken zonder interactie met gebruiker
-resultaat kan gestuurd worden naar admin via email
-gecontroleerd door: **cron deamon**
    -controleert periodiek crontab-bestanden
        -/vars/spool/cron/crontabs
        -bijgehouden per gebruiker
        -gebruiker kan deze files aanpassen (entry's toevoegen)
    -1 file per gebruiker, met als bestandsnaam de username gebruiker
    -via crontab commando kan je bestanden in die map laden
    -syntax
        -minuten uren dag-van-de-maand maand weekdag commando
        -spaties gescheiden waarde
            -minuten: 0-59
            -uren: 0-23
            - dag-van-de-maand: 1-31
            - maand: 1-12
            -weekdag: 0-7 (0 en 7 = zondag)
            - 0-15 is een bereik
            - # is commentaar
            - * is niet ingevuld
            -komma gescheiden waarden: vb: om het kwartier: 0,15,30,45
            -dagen en maanden mogen ook als 3 letters: jan, sun
            -stapgrootte na bereik: 8-18/2
        -vb: 0,15,30,45 * * * * (date; echo) > /dev/console
            # 0-10 * * 6 … > /…
    -gebruikers verhinderen om cron te gebruiken:
        -cron.allow en cron.deny files in /var/spool/cron
            -indien allow file > alle gebruikers die allowed zijn moeten vermeld w
            -geen allow maar wel deny > alle gebruikers tenzij vermeld in deny
            -geen van beide > enkel root (afhankelijk van Linux dist)
    -om iets te verwijderen uit een cron bestand gebruik je cron -r

Verklaar speciale toegangsrechten (p85)

- -sticky bit (t)
  - -save text mode
  - -vroeger: houd het bestand in het geheugen geladen (zelfs na afloop programma)
    - -vroeger veel gebruikt voor populaire programma's (bv: vi)
  - -nieuw gebruik: plaatsen op een dir
    - -enkel bestanden aanpassen waarvan eigenaar, zelfs als hij recht heeft op dir
    - -ontworpen voor temp dir (iedereen mag schrijven, maar niet elkaars files aanpassen)
    - -commando: chmon u+t /tmp (op user wordt stickybit geplaatst)
    - -nieuwe versies: chmod o+t /tmp (op other wordt stickybit geplaatst)
    - -octaal: chmod 1777 /tmp
    - -ls –ld /tmp → drwxrwxrw**t** 2 root root 8704 … (t- bit ipv x bit van others)
    - -indien anderen geen x recht hebben > T-bit ipv t-bit
  - -octale toegangscode: **1000**
- -SUID en SGID toegang op uitvoerbaar file
  - -set-user-id en set-group-id
  - -proces dat file uitvoert krijgt toegangsrechten van user/groep die door SUID en SGID geset zijn
    - -niet op basis van user die file uitvoert
  - -gewone gebruikers commando's laten uitvoeren waarvoor ze normaal geen rechten hebben
    - -vb: passwd file
      - -normaal enkel leesrecht
      - -ook schrijfrecht via passwd commando (eigenaar is root)
      - -SUID is geset op dit commando > zal zich voordoen alsof root uitvoerder is
  - -octale toegangscode: **4000**
  - -SGID op dirs
    - -files aangemaakt in deze dir worden eigendom van SGID ipv gebruiker die de file made
    - -chmod g+s …
    - -octale toegangscode: **2000**

Verklaar variabelen in de shell (p52)

-gewone variabelen
-geheugenplaats, naam en waarde
-naam begint met letters of underscore, bevat letters, cijfers of underscores
-geen declaratie ,worden gecreëerd bij toekenningopdracht
-enkel strings (geen beperking op lengte)
-variabele wordt geëxpandeerd door prefix $ (inhoud opvragen)
-$1, $2, … > argumenten op commandoregel
-omgevingsvariabelen (enkel hoofdletters)
-w mee gekopieerd naar kind shell
-overzicht met commando: set
-PATH bevat lijst met dirs waar commando's w opgezocht
-HOME bevat pad naar homedir
- toekenning:
- PATH=/bin:/usr/bin:/urs/local/bin
- PATH=$PATH:/usr/locale/games
-geen spaties voor / na = teken
-geen spaties of je moet aanhalingstekens gebruiken
-shell-metatekens vermijden
-afspraak: speciale betekenis vars > HOOFDLETTERS, anders kleine letters
-inhoud var tonen: echo $a
-met commando unset verwijder je een variabele: unset a
-opsplitsen van parameters gebeurt door IFS (internal field seperator)
-bevat standaard spatie, tab, new line
-veranderen door bv: IFS=: (stelt IFS gelijk aan dubbelepunt)
-arrays
-tabel[32]="een waarde"
-declare –a tabel (niet perse nodig kan met toekenning)
-associatieve array: declare –A tabel (associatieve array moet eerst worden gedeclareerd)
-tabel=(waarde1 waarde2 waarde3)
-tabel=(waarde1 [2]=waarde2 waarde3)
-inhoud weergeven: echo ${tabel[32]} (accolades om foutieve parameter expansie voorkomen)
-alle waarden: echo ${tabel[@]} of echo ${tabel[*]} (verschil bij expansie van variabelen > spaties)
-unset tabel (hele tabel) of unset tabel[32] (1 element)
-speciale notering: foutboodschap
-echo ${var?} > inhoudvar (controle of hij is toegekend)
-uitvoer: var:parameter null or not set
-echo ${var?parameter var vergeten te setten\?}
-uitvoer: var: parameter var vergeten te setten?
-default waarden:
-echo ${var=`dit is default waarde`}
-anti default waarden
-echo ${var+`dit wordt weergegeven als var geen waarde heeft`}
-bereik van vars
-sh opent nieuwe shell
-omgevingsvariabelen worden door gegeven, andere variabelen niet
-enkel indien ze geëxporteerd zijn: export x
-kind shell kan geen waarden terug geven aan oudershell (schrijven in file)
-speciale vars
-$# (aantal / lengte), $*, $@, $0 (naam script), $? (exitstatus), $$ (proces id), $! (

Bijvraagjes:

-Bij toekenning aan variabele, waarom geen spatie voor en na het = teken
-omdat bash een assignatie anders ziet als een commando met wat parameters erachter
-hij zal zoeken achter het commando en het mss niet vinden > error
-wel vinden is toeval en uitkomst zal niet zijn wat je wil
-waarom kan variabele alleen van ouder -> kind en niet andersom (iets met processen)

Wat zijn wildcards, en hoe worden ze door de shell verwerkt? (p50)

-speciale tekens die een patroon beschrijven (soort regexp)

-*

-duid een aantal willekeurige tekens aan

-bv: echo *

-geeft opdracht tot zoeken naar alle bestandsnamen en dirs

-bestanden die beginnen met een . zullen niet worden weergegeven

-enkel indien patroon begint met .

-vermijdt problemen met . en .. dirs die in elke map voorkomen

-?

-1 willekeurig karakter

-/

-letterlijke new line

-laat je toe om een new line te printen met echo

-#

-commentaar: de rest van de regel wordt genegeerd

-bv: echo hallo # daar > hallo

-[a-z]

-1 character die aan het patroon binnen [] voldoet (hier: interval van a tot z, dus: a,b,c… z)

-betekenis van wildcards ongedaan maken door ze binnen

-enkele aanhalingstekens te zetten

-alles wordt letterlijk geïnterpreteerd

-echo '***'

-dubbele aanhalingstekens te zetten

-echo "***"

-het teken $ en \ worden wel nog geïnterpreteerd

-echo "$naam*" > zoekt alle bestanden die beginnen met de waarde in var $naam

-ene soort aanhalingstekens mag je binnen de andere plaatsen

-echo "blabla's"

-moeten niet om hele argument staan

-echo x '*' y

-tussen aanhalingstekens mag je een new line schrijven

-na interpretatie worden aanhalingstekens verwijdert > echo ziet afzonderlijke argumenten

-interpretatie gebeurd door shell, shell piped dan door naar commando (echo)

-indien echo geen bestandsnamen vindt die aan patroon voldoen: toont hij het patroon

die tabel in de inode, wat staat daarin en waar verwijzen ze naar? wat zijn die single, double... (94)

-tabel met adressen van de datablokken van het fileobject

Beschrijf de standaarduitvoer, standaardinvoer, en standaardfoutuitvoer (p41)

-bij deze 3 zijn er bestanden gekoppeld > filediscriptoren
    -kunnen w doorverbonden met file of pipe

1/ **standaard invoer**
    -gebruiker geeft informatie in via toetsenbord
    -filediscriptor: 0 (meestal verbonden met toetsenbord: terminal)

2/ **standaard uitvoer**
    -uitvoer die op het scherm verschijnt
    -filediscriptor: 1 (meestal verbonden met scherm: terminal)

3/ **standaard error output**
    -ontworpen om errors altijd op het scherm te doen verschijnen
        -zodat fouten niet komen te staan waar juiste uitvoer had moeten komen
        -verwittigen gebruiker
        -diff file1 fiel2 > diff.out
            Diff: fiel2: No such file or directory
    -filediscriptor: 2 (bijna altijd met terminal verbonden)
        -sommige commando's producteren output via 2 zelfs bij correcte uitvoering (bv: time)

-bv:
    -/usr/bin/time –p wc /usr/man/man1/* > wc.out 2>time.out
        -p optie zorgt dat uivoer in POSIX formaat uitgeschreven w
        -time w uitgevoerd, word count om alle man pages
            -uitvoer van wc gaat naar bestand wc.out en uitvoer van time (std error) > time.out
        -geen spaties toegestaan tss "2" en ">" bij 2>bestand

-Samenvoegen van 1 en 2
    -/usr/bin/time –p wc / usr/man/man1/perltoc.1 >wc.out 2>&1
        -2>&1 wil zeggen: zet std error op zelfde file als waar std uitvoer w gestuurd (volgorde!)

Wat is een "here-document"? (p43)

-invoer specifieren voor een commando direct na het commando (dus niet in apart bestand)

```
ftp –n computernaam <<EOF
        user gast wachtwoordvangast
        dir
        put file_xyz
        delete file_xyz
        dir
        EOF
```

-de standaard invoer voor dit commando w beschouwd als de lijnen tussen het sleutelwoord na de "<<" en opnieuw datzelfde sleutelwoord alleen op 1 regel.
-de invoer bevind zich direct "here" en niet ergens in een ander bestand
-$ en `…` en \ w geïnterpreteerd tenzij ze tussen aanhalingstekens w geplaatst of voorafgegaan door \

Beschrijf de werking van het filesystem fysisch. Hoe worden ≠ fysische media in het systeem beheerd? (p88)

-logische boomstructuur verdelen over verschillende fysische disks of diskpartities (devices)
    -of zelfs op andere computers
    -meestal magnetische schijf
    -bij grote systemen w takken van de boom ondergebracht in ≠ schijven of vers. partities van schijven
        -fout op 1 van disks > niet alle data aangetast
        -moeilijker onderhoudbaar
            -wat moet gebeuren indien filesystem te klein is geworden?
-mount, unmount
    -filesystemen op ≠ devices moeten gekoppeld w aan boomstructuur (gemonteerd)
        -mount commando (komt van vroeger echt monteren van disks)
        -dan pas weet kernel dat device bestaat
        -zodat 1 logsche structuur ontstaat vanaf /
    -mount –t type device dir
    -mount –t vfat /dev/fd0 floppy
        -meestal enkel toegelaten voor root
        -devide van type type inladen onder de directory dir
            -vorige inhoud van dir w onzichtbaar, na demonteren w inhoud terug zichtbaar
            -best monteren op lege dir
            -mogelijke types staan in fstab of file mount, afh van OS
    -mount
        -toont alle gemonteerde filesystems
    -unmount
        -voordat je de schijf demonteerd moet je ze unmounten
        -unmount /dev/fd0
        -unmount /mnt/floppy
            -ofwel device unmounten ofwel dir unmounten
-/etc/fstab filesystem configuratiefile
    -≠ mogelijke namen: fstab, filesystems,mnttab,vstab (ook formaat wijkt soms iets wat af)
    -1 lijn per device

```
-/dev/hda3      /              ext2    defaults              1 1
 /dev/fd0       /mnt/floppy    vfat    noauto,user,suid      0 0
```

        Velden:
            1/ naam devicefile
            2/ plaats in directory structuur
            3/ type filesysteem (ext2 is std voor Linux, vfat is msdos, iso9660 voor cdrom)
            4/ opties: lijstje p90 onderaan
            5/ om de hoeveel dagen moet “dump” gebeuren (w niet meer gebruikt)
            6/ bepaald volgorde voor checken filesysteem door fsck
                -voor / is dit altijd 1
                -0 betekend dat het nooit automatisch gechecked w
    -gebruik fstab bestand
        1/ mount –a (alle devices in fstab w gemount, tenzij die met noauto)
        2/ monteren van devices vermeld in fstab is het genoeg met ofwel dir ofwel device
            -bv: mount /dev/fd0 of mount /mnt/floppy
        3/ devices met “user” optie kunnen w gemount door gewone users (ook terug unmount)
-df (diskfree) overzicht van geheugengebruik op de ≠ desvices

-Filesystem blocks
-logische blokken (512,1024,2048,…) (afh van hoe systeem dit impl.)
-grootte van blokken waarmee OS data leest / schrijft
-grote blokken > sneller (minder toegangen nodig) > minder optimaal diskusage
-layout logische blokken:
1/ boot blok
-1e sector v diskpartitie
-kan opstartcode bevatten > init OS
2/ super blok
-beschrijft satus van filesysteem, kan bevatten:
-omvang (aantal blokken), namen v filesystemen, naam volumes, aantal files die het kan bevatten, time laatste update en backup, vrij data-blokken,lijst vrij inodes
-tune2fs –l device > geeft overzicht superblok
-gereserveerde blokken > niet in te nemen door gebruiker
3/ inode lijst
-p94
-pointer naar datablok
4/ data blokken
-bevatten de eigenlijke gegevens
-1 datablok kan maar tot 1 fileobject behoren

Wat zijn signalen? Hoe worden ze verwerkt, en hoe kan een applicatie er mee omgaan? (p110)

-process killen > signaal sturen naar process: *kill yourself*
-kill –signaal PID's
-signaal kan een deze zijn: (overzicht: man 7 signal)
-1 (HUP) hangup > verbreekt connectie tss terminal en process
-2 (INT) interrupt > zelfde als ctrl+C in controlerenende terminal (break)
-3 (QUIT) zelfde als indrukken DEL-toets > neerslag van geheugeninhoud (wegschrijven?)
-9 (KILL) kill process (kan niet w opgevangen > there is no escape :p)(niet deftig wel krachtig)
-15 (TERM) terminate > zo snel mogelijk sluiten (mits opkuisen bronnen)
-24 (STOP) zelfde als ctrl+Z (jobcontrole stop)
-killed ook alle kinderen van deze shel
-user Shell killen > gebruiker zal w uitgelogd
-process niet mogelijk te killen zelfs met kill commando, 3 categorieen:
1/ zombies
-nog geen bervestiging van ouder ontvangen maar wel al bronnen opgeruimd
-neemt geen systeem resources meer inbeslag (enkel nog vermeld in processtabel)
2/ processen die wachten op niet beschikbare NFS-bronnen (Network File System)
-bestanden opvragen van server die niet aan staat (stoppen door signaal 2 of 3 te sturen)
3/ processen die wachten of device bv: terug spoelen tape-drive
-signalen opvangen in Shell
-trap rij-commands lijst-signaal-nummers
-trap `rm –f $new $old; exit 1` 1 2 15
-rij commando's tussen aanhalingstekens plaatsen
-rij w 2 keer gelezen: bij setten trap en bij uitvoeren trap
-indien niets moet gebeuren: ``als commando rij
-commando's w uitgevoerd indien een der signalen w opgevangen
-bv: p113

Bespreek device files en geef een beknopt overzicht van de andere soorten files met hun functionaliteit. (p100)

-device files
  -verzorgen input en output met alle devices
  -staan in /dev
  -je kan bestanden op een schijf aanspreken alsof de schijf een bestand is
  -2 soorten special files:
    1/ character special files
      -I/O op karaktergeoriënteerde of ruwe (onbehandelde) niet gebufferde basis
    2/ Block special files
      -waarvan de I/O in blokken verloopt (bv: disk)
      -data w getransfered in blokken van vaste lengte
    -device kan meestal op beide manieren w aangesproken (beide device files bestaan)
  -namen sterk afh. van implementatie
  -harde schijf 1 : hda
    -hda1 > 1^{e} partitie
  -floppydrive: fd
  -ls -l
    -geeft ipv omvang van de files, 2 getallen weer, w gebruikt door device manager in kernel:
      1/ major > zegt kernel om welke devicedriver het gaat > goede connectie maken
      2/ minor > zegt op welke manier driver moet gebruikt worden (parameter)
  -aanmaken door: mknode [c|b] majornr minornr filename
  -staan ook vermeld in fstab (mount bestand)
-andere soorten files:
  -regular files
    -normale bestanden, bevatten ascii-tekst of binaire info, uitvoerbare code, …
  -dirs
    -binaire file met lijst van andere fileobjecten
      -lijst bevat paren met naam van file en nummer inode
  -links (p98)
    -≠ filenaam refereren naar bestand
    -2 soorten: hard link en soft link of symbolic link
  -sockets
    -file dat w gebruikt ter communicatie tussen processen (bv: dev/printer)
  -Named pipes
    -pijplijnen die geopend werden op naam
    -meestal in /dev
    -andere manier van inter-proces communicatie

Bespreek de uitvoer van het ps-commando en bespreek de bijhorende procesattributen (p105)

-toont informatie over de processen die actief zijn
-veel opties > man pages (afh van distributie)
-hoofdingen van de uitvoer:
  -PRI: geeft freq in Hz van timeslice waarmee proces aan bod komt
  -NI: is het nice number
  -SIZE: omvang proces in virtueel geheugen
  -STAT: status (R=run, S=sleep, D=do not disturb>sleeping, T=gestopt, Z=zombie
-bv:

```
PID  TTY  TIME  CMD
6874 pts/9 0:00  ksh
6877 pts/9 0:01  csh
418  pts/9 0:00  csh
```

-complexere vorm kan ook tonen hoe commando's van elkaar afh zijn (kind van, …)
-commando top: scherm w om 5 seconden vernieuwd > geeft beeld hoe zwaar pc belast is

Bespreek logische en numerieke uitdrukkingen in de shell, herhaling, voorwaarde, selectie ...

-while

```
while commando
do
        #code die moet herhaald w
done
```

-exit status 0 > geslaagd

-tekst in het vet moet w herkend door Shell > steeds op nieuwe regel of voorafgaan door “;”

-if

```
if commando
then
        #code die voorwaardelijk moet uitgevoerd  w
elif commando
then
        #code die voorwaardelijk moet uitgevoerd  w
else
        #code die voorwaardelijk moet uitgevoerd  w
fi
```

-test

```
test uitdrukking
of
[  uitdrukking ]    #verkorte schijfwijze
of
[[ uitdrukking ]]   #extra functies: patronen, std numeriek vgl^n
```

-“!” (niet), “-a” (and), “-o” (of) (spaties tussen commando’s)

-for

```
for commando
do
        #code die moet herhaald w
done
```

-case

```
case  teststring in
        patroon1)
                #commando’s-1
                ;;
        Patroon2)
                #commando’s-2
                ;;
        …
esac
```

-shift

-doe een pop op de argumenten > verwijder 1e argument en schuif de rest 1 plaats op

"wat is het verschil tussen ; en & en wat voor invloed heeft het op de shell?"

-“&”

-toevoegen op einde van een commando om het op de achtergrond uit te voeren

-ja krijgt onmiddellijk na uitvoering controle over de command prompt

-“;”

-op einde commando toevoegen > staat toe meerdere commando’s op 1 lijn te schrijven

leg uit commandolijn, variabelen, metatekens, omleiden , groeperen, … (p31)

1/ commandolijn:
- -#woorden gescheiden door (| & ; ( ) < > space tab) met carrage-return erna (<cr>)
- -1e woord w geïnterpreteerd als commando
  - -1e woord w opgezocht in tabel interne commando's
  - -niet gevonden? > overlopen alle dirs vermeld in $PATH var
    - -zoekt op bestandsnamen
    - -gevonden > dan stop hij met zoeken en voert uit
    - -niet gevonden > "command not found" error
- -intern of extern commando
  - -intern: deel van shellproces
  - -extern: naam van binair uitvoerbaar programma of shellscript (name file is name command)
- -command completion (commando vervolledigen)
  - -vervolledigen van een commando met TAB, werkt ook bij argumenten
- -command history
  - -w bijgehouden per gebruiker
  - -pijltjes toets omhoog / omlaag
- -gereserveerde woorden
  - -speciale betekenis voor de shell
  - -! case do done elif else esac fi for function if in select then until while { } time [[ ]]

2/ opties en argumenten
- -optie: verander iets aan de werking van het commando
  - -POSIX: optie start met "-" teken en is meestal letter
  - -NEW POSIX: optie start met "--" gevolgd door een woord (duidelijker)
- -argument: geeft informatie zodat commando weet wat te doen (meestal bestandsnamen)

3/ metatekens
- -hebben speciale betekenis in de shell
- -tabel p33,34 enkele voorbeelden: spatie, >, <<string, $var, `…`, &&, [[uitd]]

4/ pipen
- -date | wc
- -uitvoer van 1e command met "|" doorpipen naar invoer 2e commando
- -kan meerdere keren na elkaar: wc < file | wc | wc -l

5/ redirecten
- -uitvoer programma doorsturen naar een bestand:
  - - date >> datefile.txt
  - - echo Hallo > hallofile.txt (">hallofile.txt" is geen argument van echo, w geinterp. door shell)

6/ commando's groeperen
- -niet gegrouped: date; who | wc
- -wel gegrouped: (date; who) | wc

Wat gebeurt er bij het opstarten van een UNIX machine? runlevels? en bij het afsluiten? (p114)

-**bootstrapping** (computer systeem opstarten en klaarmaken voor gebruik)
-bootproces afhankelijk van OS
-lezen instructies vanuit permanent gegevens opslag (ROM / firmware)
    -lokaliseren bootprogramma dat unix zal opstarten
        -bootprog. Opgeslagen op vaste plaats op bootable device (meestal blok0 op rootdisk)
            -kunnen meerdere gespecificeerd zijn (neemt de 1e die gevonden w)
            -net slim genoeg om uit te maken > benodigde devices toegankelijk of niet?
            -extra hardware controles: aanwezigheid geheugen, periferie, …
        -laad kernel in geheugen en geeft controle door
-kernel blijft in geheugen zolang pc draait, is kern OS
-init interne tabellen, extra hardware checks > drivers laden die nodig zijn
-start fork-exec > initieel proces met PID=1
    -testen integriteit van ≠ filesystemen (begin met root-filesystem)
    -init belangrijkste sub-taken + toelaten interactieve logins
-init scipts in /etc of /sbin + subdirs
    -fsck > checken filesystems
    -monteren locale schijfpartitities
    -… (p115 onderaan)
    -na al deze taken kunnen gebruikers inloggen > multiuser mode
-**runlevels**
    -0 > powerdown > veilig om spanning af te leggen
    -1 system admin modus
    -S of s single user mode
    -2 multiuser zonder netwerkmog.
    -3 full multiuser mode
    -4 nog te specifieren door admin
    -5 fimeware state > speciale onderhoudsmodus
    -6 shutdown en reboot status >
        vanuit andere levels rebooten > systeem naar lvl 0 dan terug naar oorspronkelijk
    -levels sterk afh van systeem
    -spcifificatie runlevels: /etc/inittab
-afsluiten
    -afsluiten voor: onderhoud, diagnose, aanpassen hardware, backup, …
    -acties:
        -gebruikers verwittigen (liefst eindje vooraf)
        -alle processen w TERM-signal gestuurd (aciviteiten beindigen)
        -subsystemen stilleggen via ingebouwde commando's
        -resterende gebruikers uitloggen en resterende prcessen gedood
        -geheugen w weggeschreven naar discs (geheuhen integreteit)
        -afh van soort shutdown
            -naar single user mode
            -volledige shutdown (cpu power down)
            -reboot
    -commando:
        -shutdown (30 seconden notice to users)
            -singleuser mode
                -na onderhoud beindigen door exit te typen > pc terug naar run-lvl3
        -shutdown –r +20 (na 20 minuten restart)
        -shutdown –h now (direct afsluiten)
        -shutdown –h –t 5 now (30 sec > TERM, 5 sec later > KILL)

ONOPGELOSTE VRAGEN:

bij schijfindeling, wat moet er staan indien het de enige partitie op de schijf is?

wat doet init? hard gecodeerd? sysinit? hardgecodeerd? altijd uitgevoerd? rc... files kort toelichten (p121)

script dat een filepatroon + minstens 1 dir als argument meekrijgt, schrijf alle files/dirs uit die voorkomen in die dir(s) en voldoen aan het patroon. Extra eisen: er mag geen foutenuitvoer in de shell verschijnen, dirs moeten eindigen met een "/".

Basisbestandspermissies uitleggen en toelichten. Welke zijn nodig voor de basiscommando's. (hiermee bedoelde hij een lijstje geven van commando's op files en dirs en bijschrijven welke permissies je ervoor nodig had)

Uitvoer van ps gegeven. Aan de hand hiervan procesattributen bespreken. Welke verschillende soorten processen kan je hieruit halen. (tip: de UID die weergegeven wordt op die uitvoer is de EUID (zie passwd))

Script: commando naam1 [naam2 naam3 ...] bestand
Minstens één naam en een bestand. Controleren op aantal argumenten. Controleren op bestaan en mogelijkheid lezen bestand. Dan voor alle namen kijken in een bestand mailadressen.txt met bepaalde vorm. (bv.: naam:bla:iets:mailadres:nogiets) Controleren dat de naam op de juiste plaats voorkomt in het bestand (begin v/d lijn), zoniet fout, controleren indien er een mailadres bijstaat, indien niet, fout. Als het voorkomt, inhoud bestand sturen naar mailadres m.b.v. mail commando. Als er geen fouten zijn, aantal geslaagde verzendingen tonen, zoniet, exit status = aantal fouten.

bestandseigendom bespreken + hoe veranderen (!= bestandrechten

uitvoer van de inhoud van al die init.d en rcX.d mappen, daar uitleg bij geven

mechanisme achter links, waarom gebruikt, bespreken, verschil hard en soft links...

Bespreek de **normale** toegangstypes voor file-objecten. Welke rechten met een file-object commando hebben. Hoe dienst voor bestandsbeveiliging.
bijvraag: Hoe veranderen? chmod rwx en chmod 777 (waarom 3 getallen), umask....

Op welke verschillende manieren kan je zelf commando's maken, hoe doe je dit, hoe start je ze op, argumenten. Geef op welke manieren je argumenten kan doorgeven op de commandolijn.

Bespreek links, commando, hoe zoek je ze met ls -l en links op directory's.

leg uit de commandolijn, hoe kan je groeperen, leg de bekendste metatekens uit, hoe kan je invoer uitvoeren omleiden?

op welke manieren kan een proces gestart worden, hoe leven ze,...

Gegeven: afdruk van init.d en rc.d en dergelijke. Commando: ls -Rd /etc/rc.d/
bijvragen: geef is de for-lus die het rc-script gebruikt om alle K-files af te lopen

Bespreek soorten eigenaars. Hoe kan je eigendom veranderen? Van wie is nieuw aangemaakt bestand?

oefening: maak een script dat als volgt aangroepen wordt "kopier bronbestanden doeldirectorys [patroon]"
-Zorg dat het scrip afsluiten met als error hoe je het moet gebruiken als er te veel of te weinig argumenten zijn.
-de parameters kunnen spaties bevatten, overloop met een forlus of alle doeldirectories bestaan.
-maak een tijdelijk bestand aan en sla de padnamen van alle bronbestanden er in op
-lees iedere lijn uit het tijdelijk bestand en kopier de file naar de doeldirectories, tel hierbij het aantal bestanden
-schrijf het aantal gekopieerde bestanden uit, verwijder de tijdelijke file