

## Unixi turvamudel

- Subjektideks on kasutajad. Kasutajate poolt käivitatud programmidel on täpselt samad õigused, mis kasutajal endal
- Igal kasutajal on oma numbriline ID (UID)
- Ühele kasutajale (juurkasutaja, `root`) on kõik lubatud; tema UID on 0
- Kaitstavaid objekte on mitmesuguseid: eeskätt failid ja kataloogid, aga ka soklid, välisseadmed jms
- Objektide identifitseerimine: objektid on paigutatud ühte suurde virtuaalsesse puukujulisse failisüsteemi
- Iga objektiga on seotud omanik ning loabitid, mis kirjeldavad erinevate subjektide juurdepääsuõigusi
- Kõrvalteed objektideni on blokeeritud

## Unix: loabitid

- Iga objektiga on seotud kolm õiguste komplekti:
  - Omanik — see, kes faili lõi või kellele administraator faili andis
  - Grupp — omaniku poolt määratud grupp, kellele ta saab määrata ülejäänutest erinevaid õigusi. Vaikimisi saab loodava faili grupiks omaniku primaarne grupp, kuid omanik saab seda muuta.
  - Ülejäänud — kõik peale omaniku ning antud grupi

<code>rwx</code>	<code>r - x</code>	<code>- - -</code>
omanik	grupp	muud
- Loabitid kehtivad ainult konkreetse objekti kohta ega parandu failipuus allapoole

## Unix: kasutajad ja grupid

- Administraator saab kasutajatest gruppe moodustada
- Gruppe identifitseeritakse numbrilise ID järgi (GID)
- Iga kasutajaga on seotud tema primaarne grupp
- Lisaks võib kasutaja kuuluda ka teistesse gruppidesse
- Kasutaja UID, GID ja lisa-GID'id seotakse tema protsessidega süsteemi sisse logimisel ning need päranduvad kõigile alamprotsessidele
- Kasutaja ise ei saa reeglina oma grupikuuluvusi muuta
- Grupid on kasutajale vajalikud ligipääsuõiguste realiseerimiseks
- Primaarset gruppi kasutatakse lisaks ka loodavate failide juurdepääsuõiguste määramisel

## Unix: loabitid

- Igas grupis on kolm bitti:
 

<code>r</code>	— lugemine	4
<code>w</code>	— kirjutamine	2
<code>x</code>	— täitmine	1
- `rwxr-x--` = 0750 (kaheksandsüsteem!!!)

## Unix: juurkasutaja

- Traditsioonilises Unixis on juurkasutaja (`root`) kõikvõimas, talle ei kehti mingid juurdepääsupiirangud
- Tohib kasutajaid lisada, muuta, kustutada
- Tohib failide omanikke muuta
- Tohib grupikuuluvusi meelevaldselt muuta
- Tohib jooksva protsessiga seotud kasutajainfot (UID, GID) muuta (login!)
- Tohib failisüsteeme monteerida, võrku konfigurereida, ...
- ⇒ Juurkasutaja peab olema usaldatav

## Unix: set\*id programmid

- Kuna sisuliselt on tegemist lüüsidega kõrgemale turvasemele, on tegu turvakriitiliste programmidega
- Kõigist set\*id programmidest me ei pääse, aga üleliigsete süsteemis hoidmine on pahanduse kohale kutsumine
- Neis programmides on sageli vaja ka tegeliku käivitaja infot, seega ei saa me käivitaja UID'd lihtsalt programmi omaniku UID-ga asendada
- Lahenduseks on tegelikult kaks UID-d (päris UID ja efektiivne UID), sama GID'i puhul

## Unix: õiguste delegeerimine

- Probleem: kasutajate õigused ainult vähenevad ülalt alla, mõne operatsiooni jaoks on vaja ajutiselt õigusi lisada
  - Näide: paroolivahetus
  - Kasutajate info (nimi, UID, GID, autentimisinfo, ...) on kirjas mingis failis
  - See fail ei saa olla kõigile kirjutatav
  - ⇒ sinna kirjutamiseks on vaja ajutiselt (range kontrolli all!) juurkasutaja õigusi
- Lahendus: *setuid*-programmid — töötavad programmifaili omaniku õigustes
- Sarnaselt on olemas ka *setgid*-programmid — töötavad programmifaili grupi poolt määratud GID'iga

## Unix: efektiivne UID

- *Setuid*-programmil muudetakse efektiivne UID (*setgid*-programmil efektiivne GID)
- Tegelikult mõjutab objektidele juurdepääsu efektiivne UID — EUID
- Analoogselt on olemas efektiivne GID — EGID
- *Setuid*-root (s.t. EUID = 0) programm saab edasi juba UID, EUID, GID, EGID muuta

## Unix: veel loabitte

00001 maailmale täitmine  
 00002 maailmale kirjutamine  
 00004 maailmale lugemine  
 00010 grupile täitmine  
 00020 grupile kirjutamine  
 00040 grupile lugemine  
 00100 omanikule täitmine  
 00200 omanikule kirjutamine  
 00400 omanikule lugemine  
 01000 "kleepbitt"  
 02000 setgid  
 04000 setuid

...

## Unix: tänapäev

- Esimene suund: tuua sisse pääsuloendid (ACL), et süsteemi paindlikumaks teha
- Olemas enam-vähem standardne liides (`setfacl`, `getfacl`), realiseeritud esialgu kommerts-Unixites, praeguseks levinud ka vabadele Unixitele
- Näide:
 

```
setfacl -m user:varmo:rw-,mask:rw- file.txt
```
- Teine suund: elimineerida kõikvõimas juurkasutaja ja asendada ta rea vähemate õigustega kasutajatega, kellest mõni suudab kasutajaid administreerida, mõni tarkvara paigaldada jne.
- Enamasti kasutatakse selleks voliloendeid
- Sellest on välja kasvanud rida Trusted-\* nimelisi süsteeme

## Unix: umask

- `umask` — bitimask nendest õigustest, mida uutel failidel kindlasti olla **ei tohi**
- `umask` pärandub alamprotsessidele
- Näide:
  - Olgu protsessil `umask 027`
  - Protsess teeb `open("file.txt", O_RDWR, 0666)`
  - `umask 0` puhul tekiks fail õigustega **rw-rw-rw-**
  - Nüüd aga tekib fail õigustega 640: **rw-r-----**

## PAM (Pluggable Authentication Modules)

- Meetod autentimisega tegelevate alamprogrammide ühte kohta koondamiseks
- Seni vajasime igale paroolle kontrollivale programmile `setuid`-lippu, see pole mõistlik
- Koondame autentimise spetsiaalse teegi sisse, mis kasutab vajadusel välist `setuid`-programmi
- Tegelikult saavutame rohkem — tsentraalse koha kasutajate autentimiseks ja ülesüsteemseteks piiranguteks
- Iga autentimist vajava teenuse jaoks saame valida mingi hulga moodulite vahel

## PAM moodulid

- Nelja liiki mooduleid:
  - *auth* — autentimismeetodi valik
  - *account* — kasutajakonto piirangud
  - *session* — konkreetse sessiooni autentimine ja piirangud
  - *password* — paroolipoliitika pealesurumiseks
- Iga moodul tagastab ühe järgmistest väärtustest:
  - *success*, *failure*, *ignore*
- Mooduli atribuudid: üks järgmistest
  - *required*, *requisite*, *optional*, *sufficient*, *binding*

## Windows NT turvamudel

- Subjekte (kasutajaid ja gruppe) identifitseeritakse süsteemisiseselt turvaidentifikaatori (SID — *Security ID*) järgi
- SAM (*Security Account Manager*) — peab kasutajate andmebaasi ja autendib kasutajaid
- Sisse logimisel jäetakse kasutaja kohta meelde tema SID ning kõigi gruppide SID'id, kuhu see kasutaja kuulub
- Objekte on palju erinevaid: failid, kataloogid, seadmed, registrivõtmed, protsessid, protsessidevahelised torud, ...
- Kõigile objektidele lisatakse ACL (nimekiri kasutajate, õiguste ja lubatavuse kolmikutest)
- Lisaks on olemas kasutajate õigused — süsteemi poolt defineeritud õigused, mille väljendamiseks pole konkreetset objekti (võrgust sisse logimise õigus, kella seadmine, ...).

## PAM konfiguratsiooni näide

```
auth    required pam_unix.so nullok_secure
auth    required pam_nologin.so
```

```
account required pam_unix.so
```

```
session required pam_unix.so
session required pam_limits.so
session optional pam_motd.so
```

```
password required pam_unix.so nullok obscure min=4 max=8
password required pam_cracklib.so retry=3 minlen=6 difok=3
```

## Windows NT: volitatud serverid

- Kasutaja poolt käivitatud protsessid töötavad kasutaja turvakontekstis (kasutaja õigustes)
- Kaitstud serverid töötavad operatsioonisüsteemi sees ning suhtlevad kasutajaprotsessidega (näide: Win32 alamsüsteem)
- Kaitstud server võib vajada kliendi turvakonteksti, et teha operatsioone kasutaja eest
- Selleks võib serverprotsess (täpsemalt serverprotsessi lõim) ajutiselt kliendina esineda (*impersonation*)

## Windows NT: ACL

- Iga objektiga seotakse turvainfo: omaniku SID, grupi SID, ACL ja süsteemne ACL (auditeerimiseks)
- ACL'i saab muuta omanik, süsteemset ACL'i administraator
- Eristatakse harilikke (failid) ja konteinerobjekte (kataloogid)
- Konteineritesse loodavad objektid pärivad oma ACL'i vaikumisi konteinerilt
- ACL koosneb järjestatud juurdepääsukirjetest (ACE — *Access Control Entry*)
- Tühi ACL tähendab igasuguste juurdepääsuõiguste puudumist
- Puuduv ACL lubab igasuguse juurdepääsu
- ACL'idega määratavad õigused sõltuvad kaitstava objekti tüübist — failil omad, printeril omad

## Windows NT: ACL läbivaatus

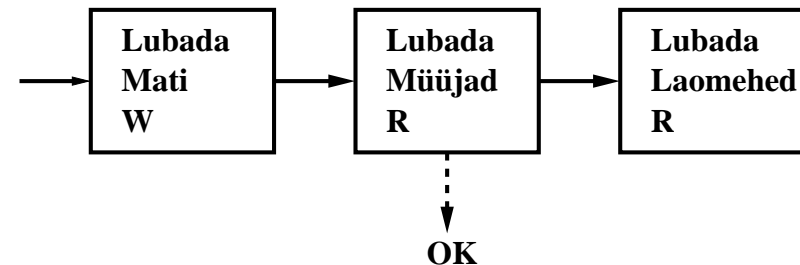
- Olgu meil kasutajaprogrammi päring, mis soovib saada mingite etteantud õigustega juurdepääsu
- ACL läbitakse kirjehaaval, kuni saadakse teada lõppvastus
- Kui avastatakse mõnda soovitud õigust keelav kirje, siis on tulemus negatiivne
- Lubavad kirjed erinevatele õigustele täiendavad üksteist ("liituvad")
- Kui kõik soovitud õigused leitakse, on tulemus positiivne
- Kui jõutakse niisama ACL lõpuni, on tulemus negatiivne
- Lisaks on spetsiaalsete lippudega ACE-d päritavate õiguste kohta

## Windows Vista: terviklustasemed

- Alates Windows Vistast on olemas MAC alged: protsesside usaldustase (kõrge/keskmine/madal) ning turvatavate objektide terviklustase
- Turvatavad objektid: failid, registrivõtmed, kasutajaliidese objektid (aknad aknateadete saatmiseks)
- Põhiline kasutus sama kasutaja erinevate protsesside isoleerimiseks
- Esimeseks suuremaks kasutuseks Internet Exploreri kaitstud mood — suurem osa IE-st jookseb madalal turvasemel ja ei saa muule süsteemile ligi
- Terviklustaseme infot säilitatakse turvatavate objektide süsteemses ACL-is, info puudumisel arvestatakse keskmist terviklustaset

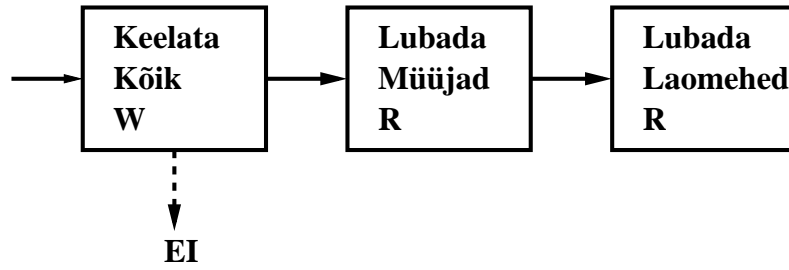
## Windows NT: ACL näited

Päring: kasutaja Mati grupist Müüjad soovib õigusi RW



## Windows NT: ACL näited

Päring: kasutaja Mati grupist Müüjad soovib õigusi RW



## Windows NT: ACL näited

Päring: kasutaja Mati grupist Müüjad soovib õigusi RW

