SISUKORD

C.2 Operatsioonisüsteemid

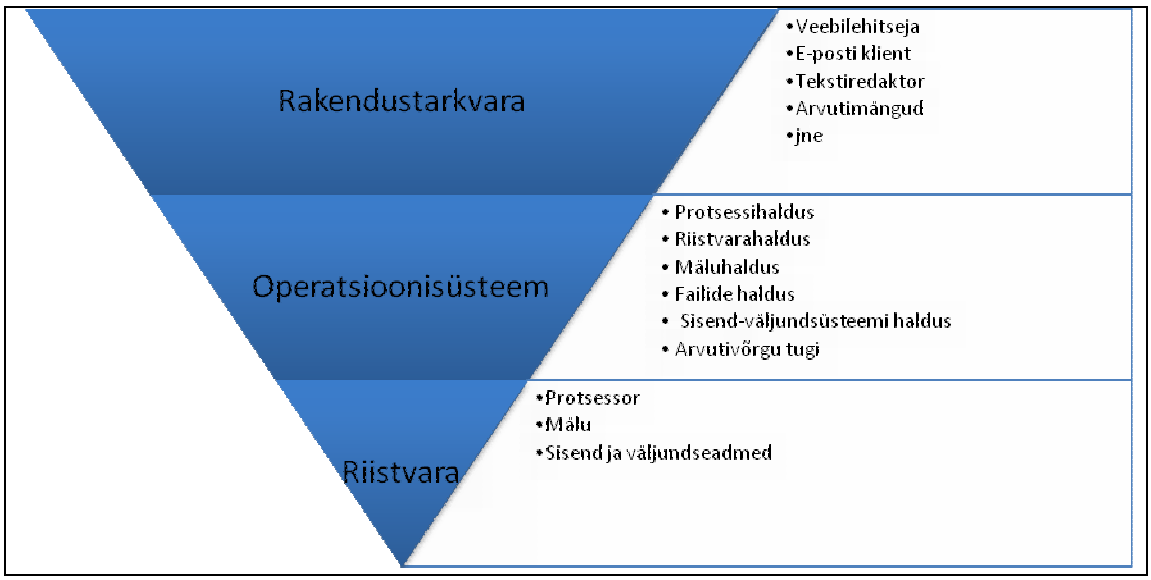
C.2.1 Põhimõtted

Eesmärgid:

- Kirjeldada tüüpilise operatsioonisüsteemi funktsioonid
- Kirjeldada operatsioonisüsteemide eri tüübid (ajajaotus-, reaalaja-, pakksüsteem)
- Kirjeldada rakendusprogrammiliidese mõiste
- Kirjeldada arvutisüsteemi ressursside haldus tarkvara abil

Mõtle!
Miks on operatsioonisüsteem üldse vajalik?
Mida operatsioonisüsteem teeb?
Mis on protsessor, mälu ja sisend-väljundseadmed?

Kõige lihtsam operatsioonisüsteemi definitsioon on: operatsioonisüsteem on vahekiht riistvara ja rakendusprogrammide vahel. See tähendab, et tänu operatsioonisüsteemile on võimalik rakendusprogramme valmistada ilma, et peaks arvestama riistvara erisusi, kõik vajalikud ressursid (protsessori tööaeg, mäluruum, sisend-väljund pordid ja seadmed jne) on operatsioonisüsteemi kontrolli all ning operatsioonisüsteem jagab neid vastavalt vajadusele erinevatele rakendusprogrammidele. Selleks, et see võimalik oleks peab operatsioonisüsteem sisaldama kindlasti protsessihaldust, riistvarahaldust, mäluhaldust ja kontrollima sisendit-väljundit. Operatsioonisüsteem võib sisaldada ka graafilist kasutajaliidest, kuid ei pea seda sisaldama: tihti on nii, et ühel operatsioonisüsteemil on võimalik kasutada mitut erinevat kasutajaliidest ning osadel operatsioonisüsteemil puudub kasutajaliides sootuks.



Operatsioonisüsteem ehk opsüsteem (inglise keeles *operating system*, lühend OS) on arvuti süsteemitarkvara, mis käivitatakse arvutis alglaadimisprogrammi poolt ning mis juhib arvutisüsteemi tööd ja teenindab rakendusprogramme. Rakendusprogrammid saadavad operatsioonisüsteemile nõudeid mitmesuguste teenuste järele läbi rakendusliideste. Kasutajad saavad vahetult suhelda opsüsteemiga madala ja rakendustaseme programmeerimisliideste kaudu ning läbi käsuinterpretaatori, kasutades selleks käsurealt ohjekeelt või graafilist kasutajaliidest. www.wikipedia.org

Tavakasutaja ei taju tihti operatsioonisüsteemi vajadust. Samas on operatsioonisüsteemi olemasolu hästi arusaadav näiteks programmeerijatele: kui operatsioonisüsteemi ei oleks, siis peaks iga uue rakenduse looja rakenduses ära kirjeldama ka riistvara iseärasused, protsessori, mälu ja erinevate komponentidega suhtlemise korra jne. Iga riistvaras tehtav muutus peaks kajastuma ka igas rakenduses ehk piltlikult öeldes: kui operatsioonisüsteemi ei oleks, siis iga riistvaras tehtava muutuse korral vajaksime sellele riistvarale sobivat tarkvararakendust, sest vana versioon enam lihtsalt ei sobiks (sest vanas versioonis on ära kirjeldamata kord, kuidas uue riistvaraga suhelda tuleb).

Operatsioonisüsteem sisaldab kindlasti protsessihaldust, riistvarahaldust, mäluhaldust, failihaldust, arvutivõrgu tuge, sisend-väljundsüsteemi haldust ning turbevahendeid.

C2.1.1 Protsessihaldus

Programm muutub protsessiks siis, kui ta käivitatakse: temast tekitatakse töötav ehk aktiivne koopia. Protsess vajab tavaliselt tööks mingi koguse protsessori tööaega, mäluruumi ning pöördub tavaliselt ka erinevate sisend-väljundseadmete poole.

Protsess on töötav programm ehk programmi aktiivne koopia

Toome lihtsa näite tekstiredaktori (tekstiredaktor on näiteks Microsoft Word, OpenOffice.org Writer jne.) baasil: kui me käivitame tekstiredaktori, siis selleks, et programm üldse käivituks peab protsessor tegema mingi hulga arvutusi (ehk tegevusi, mis on vajalikud juba ainuüksi protsessi loomiseks ning hilisemaks tööshoidmiseks). Kindlasti hõivab töötav protsess ka mingi koguse mälu: protsess on programmi aktiivne koopia ja seega peab seda kusagil ka hoidma. Samuti vajab rakendus mingisugust ligipääsu sisend-väljundseadmetele.

Tekstiredaktori puhul peavad protsessini jõudma näiteks klaviatuurilt tekstiredaktorisse sisestatud tähed. Tekstiredaktori poolt saadetav info peab omakorda kindlasti jõudma videokaartini ning sealt monitorile. Ilmselt peab tekstiredaktor sisaldama ka võimalust tehtud töö salvestada, seega vajab see protsess ka võimalust suhelda kõvaketta või mõne muu salvestusseadmega. Klaviatuur, videokaart, kõvaketas, need on ainult mõned näited sisend-väljundseadmetest.

Tänapäeva operatsioonisüsteemid võimaldavad töötada mitmel protsessil korraga. See tähendab aga seda, et operatsioonisüsteem peab tagama kõikidele protsessidele vajalikele ressurssidele juurdepääsu ning kuna ressursid arvutis on piiratud, siis tuleb ressursse erinevate protsesside vahel jagada. Samas tuleb tagada ka see, et töötavad protsessid üksteist segama ei hakkaks. Seepärast ongi üks operatsioonisüsteemi tähtsamaid ülesandeid protsessihaldus.

Protsessihalduse käigus jälgitakse, et kõik protsessid töötaksid, et neil oleks ligipääs riistvararessurssidele (kui neile vastav ligipääs on lubatud), et protsessid üksteist ei segaks ning et kõik vajalikud protsessid saaksid omavahel suhelda.

Kaasaegsetes operatsioonisüsteemides on võimalik määrata protsesside olulisust ning õiguseid. Reegel on selline, et operatsioonisüsteemile olulisemad protsessid võivad hõivata suurema hulga erinevaid ressursse, kuid on ka erandeid. Näiteks: kui meil on arvuti kuhu on paigaldatud andmebaasiserver ja veebiserver ning andmebaasiserver on antud olukorras missioonikriitiline, sest seda kasutab hulk rakendusi teistes serverites. Seega võib serverihaldur määrata, et andmebaasisesrveriga seotud protsessid on olulisemad ning neile tuleb jagada kõige suurem hulk riistvaralisi ressursse. Kriitilistel hetkedel, kui serveril on suur koormus, antakse sellisel juhul valdav osa riistvara ressursist andmebaasiserveri käsutusse, see võib tähendada, et mingil hetkel ei ole veebiserver üldse kättesaadav, kuid samas on tõenäosus, et andmebaasiserver töötab, sellevõrra suurem.

C2.1.2 Mäluhaldus

Muutmälu ehk operatiivmälu ehk põhimälu ehk suvapöördusmälu (ka: RAM (inglisekeelne lühend sõnadest *random access memory*) on digitaalseadmetel mälu, kust saab andmeid lugeda, kustutada ja kuhu saab andmeid juurde kirjutada. Operatiivmälu sisu on reeglina ajutise iseloomuga, see tähendab, et kui seadmel vool välja lülitada , siis läheb kogu info kaotsi. Muutmälu meediavormingutena on näiteks arvutis muutmälu plokid.
et.wikipedia.org

Kuna arvutis on piiratud hulk operatiivmälu, tingimustes kus töötavaid protsesse on rohkem kui üks ja kus kõik protsessid vajavad töötamiseks mingit hulka operatiivmälu tuleb operatiivmälu erinevate protsesside vahel jagada. Mäluhaldusega tuleb tagada, et iga töötav protsess saab oma kasutusse vajaliku hulga mäluruumi ning vajadusel suunab mäluhaldur vähemkasutatavad protsessid operatiivmälu asemel kasutama saalemälu.

Saalemälu ehk virtuaalmälu on võimalus, kus operatiivmälu laiendusena kasutatakse ära osa kõvakettaruumist. Sellisel juhul jäetakse protsessile mulje, et vajalik info asub operatiivmälus, tegelikult kirjutatakse ja loetakse seda hoopis kõvakettalt. Erinevates operatsioonisüsteemides on see funktsioon realiseeritud erinevalt: UNIXi laadsetes operatsioonisüsteemides kasutatakse selleks reeglina spetsiaalselt selleks otstarbeks eraldatud kõvaketta loogilist osa ehk partitsiooni. Microsoft Windows operatsioonides samal otstarbel kasutuses eraldi fail või failid.

Ressursside haldamist teostatakse tihti kasutades riistvaralisi vahendeid: näiteks on riistvaras realiseeritud võimalus kaitsta erinevate protsesside poolt hõivatud mäluruumi nii, et teised protsessid sinna ligipääsu (või kirjutamisõigust) ei oma, samuti on saalemälu tugi riistvaraline jne. Lihtsustatult võib seega öelda, et kasutaja andmed on tihti operatsioonisüsteemi poolt kaitstud kasutades riistvaralisi vahendeid.

C2.1.3 Failihaldus

Fail on sarnaste andmete kogum, mis on salvestatud tavaliselt arvuti kõvakettale eraldiseisva üksusena, ning mida töödeldakse arvutis tervikuna. Näiteks, tekstifail koosneb tekstist, aga programmifail (nt exe) koosneb arvuti jaoks täidetavatest käskudest (ja ka programmile vajalikest andmetest).
et.wikipedia.org

Operatsioonisüsteem vastutab failide loomise, kustutamise ja muutmise eest. Tavaliselt on selleks operatsioonisüsteemis realiseeritud ühe või rohkema failisüsteemi tugi. Failisüsteem määrab ära hulga andmete salvestamiseks, kustutamiseks ja muutmiseks vajalikke parameetreid: millisteks loogilisteks osadeks on kõvaketas jaotatud, kuidas andmeid kõvakettale salvestatakse, mis on fail, mil viisil on võimalik asukohad struktureerida jne. Erinevad operatsioonisüsteemid kasutavad erinevaid failisüsteeme ning ühe operatsioonisüsteemi jaoks vormindatud kõvaketas ei pruugi olla loetav (ning sinna pole siis võimalik ka kirjutada) teise operatsioonisüsteemi poolt.

Failisüsteem on viis arvutis andmefailide haldamiseks ja talletamiseks. Failisüsteemid asuvad tihti otse mõne mäluseadme peal, näiteks nagu kõvaketas või CD-ROM. Samas on olemas ka failisüsteeme, mille andmeid ei hoita kohalikus arvutis, näiteks võrgufailisüsteemid NFS ning SMB. Lisaks sellele eksisteerivad ka täiesti virtuaalsed failisüsteemid, mille sisu genereeritakse konkreetsel kujul ainult hetkel, mil sealt andmeid loetakse — näiteks Linuxi /proc failisüsteem
et.wikipedia.org

Failidega seotud operatsioonid on tavaliselt operatsioonisüsteemi funktsioonid ning seepärast ei pea rakenduse programmeerija vastavaid vahendeid ning funktsioone looma vaid saab neid oma rakenduse loomisel kasutada.

C2.1.4 Sisend- ja väljundseadmete haldus

Peamiselt mõeldakse sisend-väljundseadmete halduse all seda, et operatsioonisüsteem jagab sisend-väljundseadmete ressursse erinevate rakenduste vahel, reageerib seadmete genereeritud katkestustele ning teostab katkestuse puhul ettenähtud tegevused.

Kõik kaasaegsed operatsioonisüsteemid on PnP (*Plug and Play*) tehnoloogia toega ning vastava riistvara korral suudab sisend-väljundseadmetele jagada riistvaralisi süsteemseid ressursse (katkestused, mäluaadressid, sisend-väljud aadresse jne). Seega peab operatsioonisüsteem lisaks sellele, et jagada protsessidele ligipääsu sisend-väljundseadmetele, tagama ka sisend-väljundseadmete töö.

C2.1.5 Arvutivõrgu tugi

Kuna arvutivõrk on arvutikasutajatele tänapäeval väga oluline, siis tihti öeldakse, et arvutivõrgu tugi on operatsioonisüsteemi üks nõutavatest omadustest. Lihtsalt öeldes tähendab see seda, et tänapäeval oskab iga kaasaegne operatsioonisüsteem vastava riistvara olemasolul suhelda teiste arvutivõrguga ühendatud arvutitega ning kasutada enamlevinud võrguteenuseid.

Iga kaasaegne operatsioonisüsteem omab TCP/IP protokolli tuge ja on seega võimalik ühendada sellise operatsioonisüsteemiga varustatud arvutit internetiga. Samuti on operatsioonisüsteemis tihti kaasas hulk rakendustarkvara (veebilehitseja, e-posti klient, jne), mille abil on võimalik kasutada enamlevinud võrguteenuseid. Enamlevinud operatsioonisüsteemid sisaldavad ka tarkvaralist tulemüüri ning mõne enamlevinud võrgufailisüsteemi tuge.

Tulemüür on tarkvara või seadeldis, mis piirab ja reguleerib võrguliiklust arvutivõrgus vastavalt seadistatud piirangutele.
et.wikipedia.org

C2.1.6 Operatsioonisüsteemide liigitus

Operatsioonisüsteeme liigitakse vägagi erinevalt, kuid toome siinkohal ära mõningad tüübid.

Paketttöötlus-operatsioonisüsteemides ei suhtle kasutaja arvutiga otse, vaid operaator valmistab ette paketid, mida töödelda. Pakett-töötlus operatsioonisüsteeme on ka multiprogrammilisi, see tähendab et vähendamaks protsessori jõudeaega on töös korraga mitu paketti. Paketttöötlus operatsioonisüsteeme personaalarvutites ei kasutata.

Ajajaotus-operatsioonisüsteemides käib protsessidele ressursside jagamine nii, et täitmisel olevaid protsesse vahetatakse hästi kiiresti, seepärast jääb kasutajale mulje, et kõik protsessid ehk kogu süsteem töötab. Ajajaotus muutub keerulisemaks, kuid efektiivsemaks, kui arvutisüsteemis on kasutusel mitmetuumaline protsessor, sellisel juhul on võimalik täita rohkem kui ühte protsessi üheaegselt Ajajaotus operatsioonisüsteem on interaktiivne süsteem, sest tagab otsese suhtluse kasutaja ja programmi vahel.

Ajajaotus-operatsioonisüsteemis on igal protsessil kolm võimaliku olekut: ootel (*waiting*), valmis (*ready*) ja töös (*running*). Ootel on üks protsess siis, kui protsess ootab sisendandmeid milleta ei saa protsess jätkata (klaviatuurilt, kõvakettalt jne loetavad andmed). Valmis on üks protsess siis, kui kõik vajalikud eeldused on täidetud ja protsess on võimeline jätkama (ootab, et protsessihaldur jagaks talle vajalike ressursse).

Reaalaja-operatsioonisüsteemi iseloomulik omadus on, et operatsioonisüsteem peab andma kindla aja jooksul õige vastuse. Selliseid operatsioonisüsteeme on omakorda kahte liiki: kõvad (hard) – garanteerib õigeks ajaks töö valmimise ja pehmed (soft) – kriitilised protsessid saavad suurema prioriteedi.
Reaalaja operatsioonisüsteemid on enamasti kasutusel arvutisüsteemides, mille peamine ülesanne on juhtida mingisuguseid väliseid seadmeid ja protsesse ning kus on esmatähtis reageerimise aeg.

Paralleeloperatsioonisüsteemid on mõeldud mitme protsessoriga arvutisüsteemidel kasutamiseks. Sellised arvutisüsteemid on omavahel tugevalt seotud (neil on ühine mälu, IO, jne) ning nende kasutamise peamine eesmärk on saavutada oluliselt suurem jõudlus, kui seda on ühe protsessoriga arvutisüsteemidel. Samuti on sellisel juhul süsteemil ka suurem tõrketaluvus, kuid siiski ei ole see omadus nii oluline kui jõudluse kasv. Mitme protsessoriga arvutisüsteem on odavam kui mitu süsteemi eraldi.

Hajusoperatsioonisüsteemid on operatsioonisüsteemid, mis töötavad arvutisüsteemidel, mis võivad paikneda suurel maa-alal. Enamasti tähendab see seda, et erinevad arvutisüsteemiosad on omavahel ühendatud arvutivõrgu abil ning jagavad arvutivõrgu abil üksteisega erinevaid ressursse. Tuntum selline lahendus on klient-server arvutisüsteem.

Personaalarvuti operatsioonisüsteemid tekkisid 70'ndatel, algselt polnud ei multikasutaja ega multitegumi tuge. See tähendab, et personaalarvuti operatsioonisüsteemides oli algselt ainult üks kasutaja ja korraga töötas vaid üks protsess, põhitähelepanu pöörati kasutamismugavusele, turvalisus ja arvutivõrgutugi oli teisejärguline ehk reeglina puudus. Tänapäeval on see paradigma oluliselt muutunud: kõik kaasaegsed personaalarvutites kasutatavad operatsioonisüsteemid on multikasutaja toega. See tähendab seda, et operatsioonisüsteemis on olemas rohkem kui üks kasutaja ning on võimalik, et korraga kasutab operatsioonisüsteemi rohkem kui üks kasutaja. Samuti on pikka aega personaalarvutites kasutatavates operatsioonisüsteemides kasutusel multitegumi tugi, mis tähendab seda, et mitu protsessi töötab üheaegselt, see on tavaliselt teostatud ajajaotuse abil. Kuna kaasaegses arvutis on tihti kasutusel mitmetuumalised protsessorid ning võib juhtuda et selliseid protsessoreid on suisa mitu, siis on personaalarvutites kasutatavad operatsioonisüsteemid paralleeloperatsioonisüsteemide omadustega.
Kaasaegsetes personaalarvutite operatsioonisüsteemides pööratakse suurt tähelepanu turvalisusele ning juba pikka aega on nõutav, et operatsioonisüsteemil oleks sisseehitatud arvutivõrgu tugi, reeglina on see tugi TCP/IP protokollil baseeruvatele võrkudele, vanemad operatsioonisüsteemid toetavad ainult IPv4 protokollistiku, uutel on aga ka IPv6 tugi.

Pihuarvuti ehk mobiilsete seadmete operatsioonisüsteemid sarnanevad paljus kaasajal personaalarvutites kasutatavatele operatsioonisüsteemidele, kuid nendes on tihti realiseeritud väiksem hulk funktsioone. Arvutisüsteem koosneb alati piiratud hulgast ressurssidest (piiratud hulk protsessori tööaega, piiratud hulk mäluruumi jne), kuid mobiilsete seadmete operatsioonisüsteem peab arvestama hulga lisanduvate nõudmiste ja piirangutega: protsessori jõudlus, mälu hulk, salvestusruum on veelgi piiratumad (väiksemad). Äärmiselt oluline sellistes operatsioonisüsteemides ühendatavus- ehk suhtlusvõimalused välismaailmaga ning kiirus on teisejärguline.
Kaasaegsete mobiilsete seadmete operatsioonisüsteemide kasutusala on lisaks pihuarvutitele ka mobiiltelefonid. Kuna mobiiltelefonide kasutajate nõudlus erinevate rakenduste järgi on viimastel aastatel oluliselt kasvanud, siis on mõistlik kasutada ka mobiiltelefonides kindlaid operatsioonisüsteeme. Nii on tarkvaraarendajatel lihtne toota erinevaid rakendusi.

C2.1.7 Rakendusprogrammiliidese mõiste

Üks operatsioonisüsteemi peamisi funktsioone on luua programmeerijale ühtne platvorm rakendustarkvara loomiseks. Kui nüüd piltlikult öelda, siis peamine probleem on, et programmeerijatel peab olema võimalus luua rakendustarkvara arvestamata riistvara iseärasusi. Selleks, et seda saavutada on loodud rakendusprogrammliides ehk API (*Application Programming Interface*).

Rakendusprogrammliides on kirjelduste (teenuste) kogum, mis kirjeldab riistvaraga suhtlemise korra, peites kasutaja ja programmeerija jaoks ära riistvara omapärad

Lihtsalt öeldes tähendab see seda, et programmeerijal tuleb öelda näiteks, et "tahan printida" ning kui see tegevus on rakendusprogrammliideses kirjeldatud, siis on juba

rakendusprogrammliidese ülesanne see toiming seadmeajureid (*driver'id*) kasutades riistvara abil teostada.
Erinevate operatsioonisüsteemide rakendusprogrammliidesed on tihti teostatud erinevalt ning osalt just seepärast ei ole võimalik vahel ühe operatsioonisüsteemi jaoks kirjutatud rakendustarkvara teises operatsioonisüsteemis kasutada.

C2.1.8 Riistvara haldus tarkvara abil

Kaasaegsed operatsioonisüsteemid peavad arvet arvutisüsteemis leiduva riistvara osas ning genereeritavate katkestuste puhul tehtavad tegevused on realiseeritud paljus tarkvaras. See tähendab lihtsalt öeldes seda, et kui näiteks kasutaja vajutab klaviatuuril mõnda klahvi, siis genereeritakse riistvaraline katkestuse signaal, kuid katkestusele reageering on juba paljus tarkvara ülesanne.
Samuti on seoses *Plug and Play* tehnoloogia kasutuselevõtuga süsteemsete ressursside jagamine riistvara komponentidele muutunud operatsioonisüsteemi ülesandeks.

Kaasaegses arvutisüsteemis peab riistvarakomponentide üle arvet operatsioonisüsteem ning paljudel juhtudel on sisend-väljundseadmete poole pöördumine võimalik ainult operatsioonisüsteemi vahendusel

Kirjelda!

- Miks on operatsioonisüsteem vajalik?
- Millised on operatsioonisüsteemi peamised omadused?
- Millised on operatsioonisüsteemide peamised liigid?
- Millist tüüpi operatsioonisüsteemid on kasutusel personaalarvutites?
- Mis on rakendusprogrammliides ehk API?

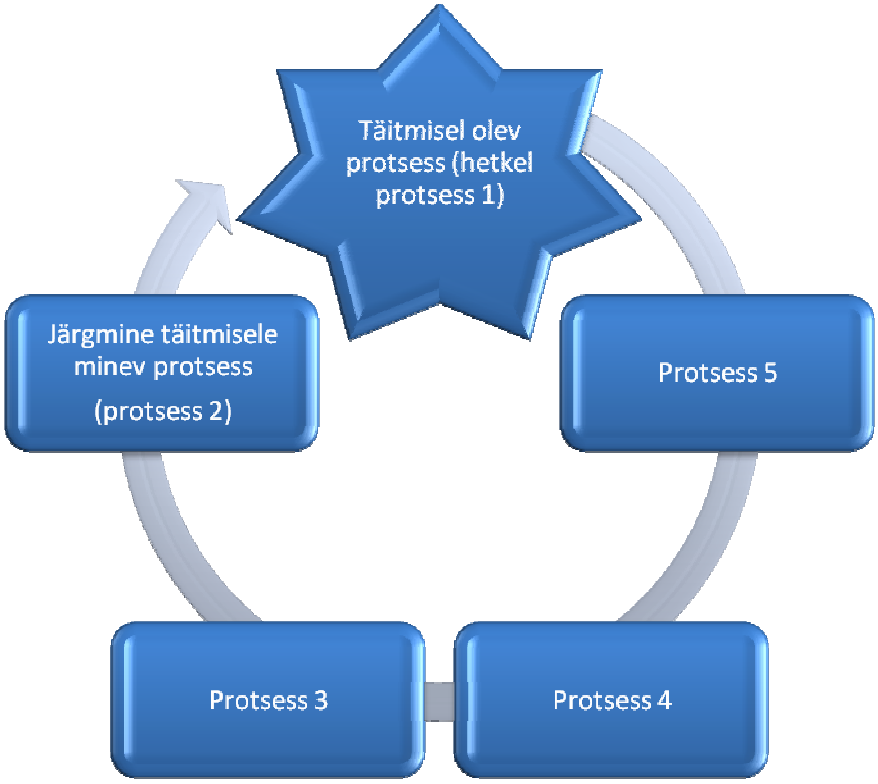
C.2.2 Üheaegsed- ja paralleelprotsessid

Eesmärgid:

- Määrata üheaegsuse olemasolu operatsioonisüsteemis
- Kirjeldada vastastikuse välistuse probleem
- Kirjeldada lõime mõiste
- Kirjeldada kontekstivahetuse mõiste

Mõtle!
Mis on protsess?
Kuidas on omavahel seotud protsess ja programm?
Mida tähendab paralleelsus ja üheaegsus?
Milliseid paralleelprotsesse sa tead?

Kõik kaasaegsed operatsioonisüsteemid on multitegumi toega ehk võimaldavad erinevatel protsessidel töötada samaaegselt. Nagu juba ajajaotus operatsioonisüsteemides selgitatud võib see tähendada seda, et igast protsessist täidetakse korraga ainult osa ja vahetatakse töösolevat protsessi piisavalt sageli, et kõik protsessid saaksid mingi aja jooksul kasutada riistvara ressursse.
Kõige lihtsamal juhul tehakse seda nii, et kõik töös olevad protsessid pannakse mingil viisil järjekorda ning seejärel määratakse ajahulk, mis igale protsessile korraga eraldatakse (ajaaken ehk *time slice*), kui aeg täis, siis vahetatakse töös olevat protsessi ning kui eelmine protsess ei lõpetanud talle eraldatud aja jooksul tööd, siis liigub see järjekorra lõppu, et oodata järgmist ajaakna eraldamist.



Et selline protsesside vahetamine võimalik oleks sisaldab iga protsess lisaks töö kirjeldusele ka protsessile eraldatud numbrit, tema aktiivset hetkeseisu (ehk seisu, kuhu protsessi

täitmisega ollakse jõutud ik *programm counter*), protsessori registrite hetkeseisu, ajutisi muutujaid (mis protsessoris on pinus) ja globaalseid muutujaid.

Tegelikkuses on protsessidele protsessoriaja jagamiseks olemas mitmeid erinevaid loogikaid:

- Lihtjärjekord (*First Come First Served)* – protsesse täidetakse nende saabumise järjekorras kuni töö valmimiseni, kui mõni protsess blokeerub, siis võetakse järgmine
- Alates lühemast (*shortest first*) – iga protsessiga seotakse järgmise protsessoriaja küsitav pikkus ning esimesena võetakse täitmisse lühiajalisema sooviga protsess
- Prioriteedi järgi planeerimine – iga protsessiga seotakse prioriteet, suurema prioriteediga protsess täidetakse esimesena, võrdsete prioriteetide korral võetakse täitmisse esimesena täitmisele saabunud protsess
- Ringiratast (*round robin)* planeerimine – igale protsessile eraldatakse korraga kindel hulk protsessori aega ja selle aja möödudes tõrjutakse täitmisest protsess välja ja pannakse järjekorra lõppu

C.2.2.1 Kopereeruvad protsessid

Üheaegseid protsesse on kahte liiki: **sõltumatud (*independent*) ja koopereeruvad (*cooperating*)** protsessid.
Sõltumatud protsessid on sellised protsessid, mis üksteist kuidagi ei mõjuta. Sellised protsessid on seetõttu täitmisel täiesti iseseisvalt ning ei mõjuta (ega ole ka mõjutatud) teiste protsesside töö tulemustest. Sõltumatuid protsesse nimetatakse tihti ka paralleelseteks protsessideks: sõltumatuid protsesse on võimalik täita paralleelselt.
Koopereeruvad protsessid on protsessid, mis võivad üksteise tööd mõjutada ning mis seepärast peavad arvestama kõigi sellest tulenevate mõjutustega.

Kui paralleelsete protsesside täitmisel ei teki erilisi probleeme, siis koopereeruvate protsessidega on olukord mõnevõrra keerulisem: need protsessid peavad küll saama omavahel suheldud, kuid endiselt peab olema tagatud erinevate protsesside andmete kaitstus teiste protsesside lubamatu pöördumise eest. Kuna arvutis on mitmeid ressursse, mille poole tohib pöörduda ainult üks protsess korraga, siis üks peamisi probleeme on **vastastikuse välistuse probleem**. Nimelt tihti määratakse ära kord, kus ühe protsessi poolt hõivatud ressurss ei ole kättesaadav teiste protsesside poolt. Toome siinkohal lihtsa näite, kui üks protsess kirjutab midagi mõnda faili, siis lukustatakse see fail kuni töö lõppemiseni ning ükski teine protsess sinna vahepeal kirjutada ei saa, selleks on operatsioonisüsteemis realiseeritud võimalus lukustada fail ühele protsessile. Samalaadse näite võib tuua ka klaviatuuri puhul: kui kaks või enam protsessi töötavad samaaegselt, siis klaviatuuri katkestuse saabumise puhul peab olema kindlaksmääratud, milline protsess reageerib saabunud infole.

C2.2.2 Lõime mõiste

Iga protsess koosneb vähemalt ühest lõimest (ik *thread*), see on peaprogramm, samas võib protsess täitmise käigus luua uusi lõimi.

Iga protsess koosneb vähemalt ühest lõimest, lõim (*thread*) on lihtsam (kergekaaluline) protsess. Protsess ja tema lõimed moodustavad tegumi (*task*)

Riistvaratootjad on muuhulgas mõelnud välja mitmeid tehnoloogiaid lõimede töötlemise lihtsustamiseks, tuntum neist on kindlasti Inteli tehnoloogia Hyper-Threading. See on tehnoloogia, mis üht füüsilist protsessorit võimaldab operatsioonisüsteemil virtuaalselt tõlgendada kahe eraldi protsessorina ning seeläbi ühel ajahetkel võtta operatsioonisüsteemilt täitmiseks kaks erinevat lõime üheaegselt. Nii saavutatakse kuni 30% jõudluse kasv.

en.wikipedia.org

C2.2.3 Kontekstivahetuse mõiste

Nagu eelnevalt kirjeldatud, vahetatakse täitmisel olevaid protsesse üsna tihedalt. Täitmisel olevat protsessi vahetades peab kindlasti salvestama eelmise protsessi hetkeseis ja laadima järgmise protsessi eelnevalt salvestatud oleku, et tööd jätkata.

Täitmisel oleva protsessi vahetust koos sinna juurde kuuluvate tegevustega nimetatakse kontekstivahetuseks.

See on reeglina üsna ajamahukas tegevus ning täpne aeg, mis kontekstivahetuseks kulub sõltub paljuski tarkvarast. Sellest tulenevalt tuleb kontekstivahetusi planeerida piisavalt harva, sest muidu võib tekkida olukord, kus operatsioonisüsteem kulutab täitmisel olevate protsesside vahetamisega samas suurusjärgus aega (või rohkemgi), kui protsesside täitmisega tegelemiseks.

en.wikipedia.org

Kirjelda!
Mis on protsess ja mille poolest see erineb lõimest?
Mis on vastastikuse välistuse probleem?

C.2.3 Mäluhaldus

Eesmärgid

- Selgitada virtuaalmälu mõiste
- Kirjeldada virtuaalmälu realiseerimise võimalused riistvaras ja tarkvaraliselt
- Kirjeldada puksimise (thrashing) mõiste
- Kirjeldada mäluhierarhia mõiste
- Kirjeldada failisüsteemi funktsioonid

Programmi võib vaadelda kui käskude ja andmete jada, mis tavaliselt asub mõnel andmekandjal (tüüpiliselt siis kõvakettal). Kui nüüd programm käivitada, siis loetakse see hulk infot mällu ning protsessor hakkab mälust käske ja andmeid lugedes protsessi täitma.

Igale protsessile eraldatud mälu jagatakse lehekülgedeks (*pages*), nii et iga lehekülg on tavaliselt 4kB. Lehekülgedel pannakse vastavusse loogilised aadressid (mäluaadressid, mida näeb ja saab kasutada protsess) ning füüsilised aadressid (mäluaadressid, mida mäluseade tegelikult näeb ja kasutab).
Tegelikult on sellise tegevuse jaoks olemas lausa riistavaline seade MMU (*Memory Management Unit*), mis tegelebki riistvara tasemel loogiliste ja füüsiliste aadresside ühendamisega. MMU on näiteks Inteli protsessoritel sisseehitatud alates protsessorist 80386.

Virtuaalmälu abil on võimalik protsessile anda virtuaalselt järjestikust mäluruumi, ilma, et see info tegelikult üldse järjestikuliset mälus asetseks.

Sellised leheküljed moodustavad omakorda segmente ning ühe protsessiga seotud mälu lehekülgi nimetatakse töökomplektiks (*working set*). Tegelikult võivad erinevad leheküljed kuuluda veel lisaks ka erinevate protsesside töökomplektide hulka.

Lehekülgede kasutamine lisab omamoodi ka turvalisust, nimelt on igal lehekülje suhtes võimalik protsessile anda kolme liiki õiguseid: lugemis-, kirjutamis- ja käivitusõigus. Nii võib juhtuda, et ühel protsessil on võimalik küll lehekülge kasutada, kuid samas ei ole võimalik protsessil teada saada, mis andmed sellel leheküljel on. Protsessid otse füüsiliste aadresside poole üldjuhul pöörduda ei tohi ning seepärast peab neid õiguseid järgima.

C2.3.1 Mälulehekülgede saalimine

Lisaks sellele, et mälu leheküljel kirjeldatud füüsilised mäluaadressid võivad asuda muutmälus, on kaasaegsel riistvaral ja operatsioonisüsteemidel võimalik kirjutada ka osa lehekülgi tegelikult hoopis kõvakettale. Seeläbi saab protsessidele jagada rohkem mäluruumi, kui arvutisüsteemil tegelikult füüsilist muutmälu üldse on. Sellist tegevust nimetatakse mälulehekülgede saalimiseks (*SWAP*).

Kui lehekülgede saalimist kasutatakse koos virtuaalmäluga, siis peab operatsioonisüsteem järge pidama kasutusel olevatel lehekülgedel ja neil lehekülgedel, mida enam ei kasutata või mis ei ole mõnda aega kasutuses olnud. Kui opsüsteemil läheb lehekülge vaja või kui mõni programm nõuab saalealale kirjutatud lehekülge, siis kirjutab OS mingi lehekülje saalealale ja toob teise lehekülje mällu. (Sellest, kuidas opsüsteem otsustab, milliseid lehekülgi sisse või välja saalida, räägib lehekülgede asenduspoliitika.) Sel moel saab kasutada rohkem operatiivmälu, kui

arvutil tegelikult on.
et.wikipedia.org

Tehnoloogia tööpõhimõte on tegelikult lihtne: kuna protsess ei tea nii kui nii kus füüsiliselt leheküljed asuvad, siis võib operatsioonisüsteem vähemkasutatavad (või kõige vanemad, oleneb kasutatavast algoritmist) leheküljed mälust kõvakettale kirjutada ning kasutada vabanenud mäluosa mingite muude andmete jaoks. Kui nüüd protsessil tekib vajadus mälust väljakirjutatud lehekülje järgi, siis tekib leheküljepöördusviga (*page fault*), seejärel otsustab operatsioonisüsteem kas selline lehekülg on olemas ning kui on tuleb leida vaba mäluruum (vajadusel mõni teine lehekülg mälust välja saalida) ning vajalik lehekülg mällu tagasi tuua. Kui vajalik lehekülg on mällu tagasi toodud, siis käivitatakse viimane käsk uuesti ning protsess saab täitmiseks vajalikud andmed.

C2.3.2 Pukslemise mõiste

Kui protsessile vajalik mälulehekülg ei asu mälus, siis juhul kui protsess selle mälulehekülje poole pöördub tekib leheküljepöördusviga (*page fault*). Leheküljepöördusvea tekkimisel jääb protsessi täitmine ootele ning vajalik mälulehekülg tuuakse mällu.

Kui mälus asuvate lehekülgede arv pole piisavalt suur, siis muutub leheküljepöördusvigade väga suureks, ning protsessori efektiivne tööaeg väheneb. Operatsioonisüsteem võib aga seda olukorda tõlgendada selliselt, et kuna protsessori aega on üle, siis võib protsesse juurde anda, see omakorda tekitab aga veel suurema leheküljepöördusvigade arvu. See võib lõppeda halvimal juhul sellega, et protsess tegeleb ainult saalimisega.

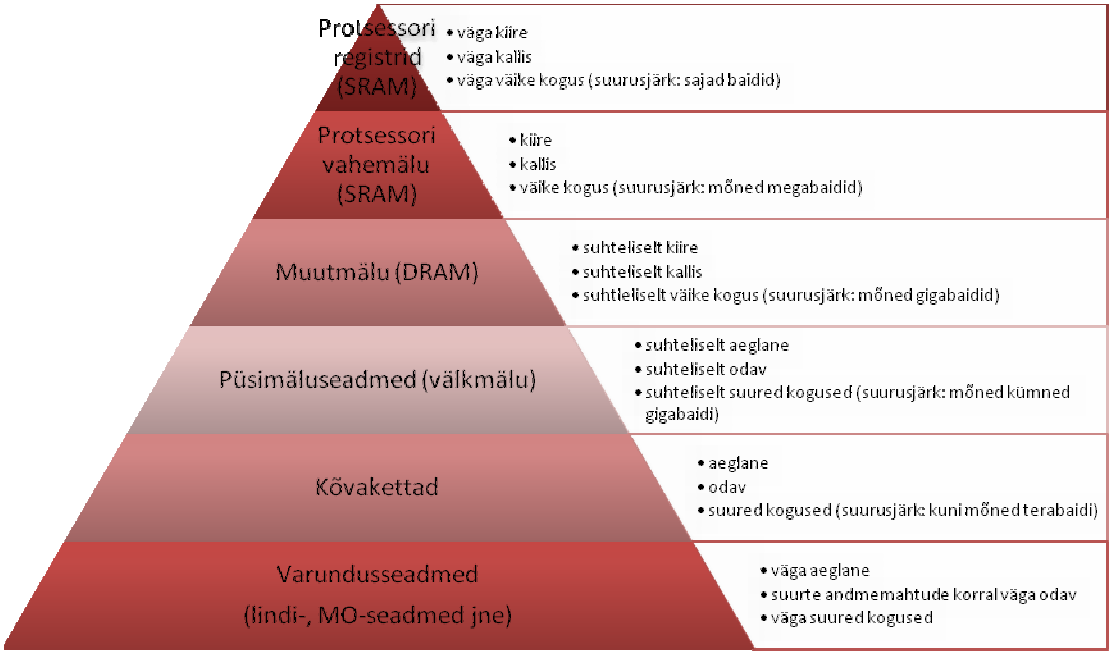
Olukorda, kus protsess tegeleb peaasjalikult mälulehekülgede saalimisega, nimetatakse pukslemiseks (*thrashing*)

en.wikipedia.org

C2.3.3 Mäluhierarhia mõiste

Mälu- ja varundusseadmete jaotust arvutisüsteemis nimetatakse mäluhierarhiaks (*memory hierarchy*). Mõte seisneb selles, et erinevatel mälutüüpidel on erinev pöördusaeg, kiirus ja hind. Mida suurem on kiirus ja väiksem pöördusaeg, seda suurem on ka mäluseadme hind ühe mäluühiku kohta (ning seda väiksem on ka sellise mälu kogus arvutisüsteemis).

Mäluhierarhia jagab mälu- ja varundusseadmed nii, et kõrgemates kihtides on seadmed mille pöördusaeg ja kiirus on suurem ning mille kogus arvutisüsteemis on väiksem.



C2.3.4 Failisüsteemi funktsioonid

Selleks, et salvestada faile kõvakettale ja muudele andmekandjatele standardiseeritud kujul, on kasutusele võetud failisüsteem. Operatsioonisüsteem käsitleb andmekandjaid, kui gruppi andmeblokke, failisüsteem kirjeldab siinjuures kui suured on blokid, kuidas andmeblokke kasutatakse, kirjeldatakse jne.

Failisüsteem on viis arvutis andmefailide haldamiseks ja talletamiseks. Failisüsteemid asuvad tihti otse mõne mäluseadme peal, näiteks nagu kõvaketas või CD-ROM, kuid on ka failisüsteeme, mille andmeid ei hoita kohalikus arvutis, näiteks võrgufailisüsteemid NFS ning SMB. Lisaks sellele eksisteerivad ka täiesti virtuaalsed failisüsteemid, mille sisu genereeritakse konkreetsel kujul ainult hetkel, mil sealt andmeid loetakse — näiteks Linuxi /proc failisüsteem.
Tänapäeva operatsioonisüsteemides on virtuaalfailisüsteemi kiht, mis seob erinevaid konkreetseid failisüsteemi rakendusi ühtseks tervikuks; sõnaga "failisüsteem" võib kirjeldada nii konkreetset failisüsteemi rakendust kui ka nende kooslust.
Tänapäeval on laialtkasutatavad failisüsteemid hierarhilise (nö. puukujulise) ülesehitusega, koosnedes üksteise sees asetsevatest kataloogidest, ning kataloogides asuvatest failidest. Ajalooliselt on kasutusel olnud ka kataloogideta, ning ühetasemelised kataloogidega failisüsteemid.
et.wikipedia.org

Lihtsalt öeldes kirjeldab failisüsteem kuidas operatsioonisüsteem haldab faile jagades neid andmeblokkidesse. Erinevad operatsioonid kasutavad erinevaid failisüsteeme, mis võivad üksteisest märgatavalt erineda nii omaduste kui ka funktsioonide poolest.

Üks füüsiline andmekandja jagatakse üheks või enamaks loogiliseks osaks ehk partitsiooniks, iga partitsioon võib olla vormindatud kasutama erinevat failisüsteemi.

Failisüsteemi peamised ülesanded on:

- Andmete hoidmine
- Failide struktureerimine
- Andmetele õigustekohase ligipääsu tagamine
- Andmete kaitse
- Erinevate andmekandjatest ühetaolise keskkonna loomine

Fail on loogiliselt grupeeritud kogus andmeid (infot).

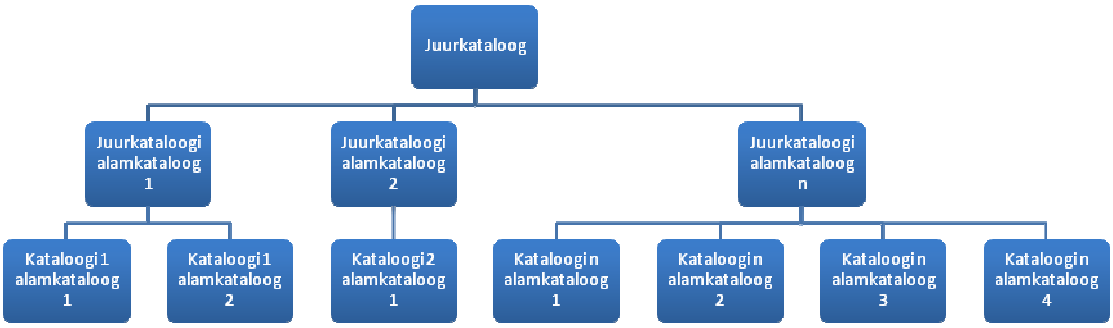
Igal failil on hulk omadusi, mis teda iseloomustavad, kõige tüüpilisemad omadused on:

- Nimi
- Tüüp
- Asukoht
- Suurus
- Ligipääsu- ja kasutusõigused
- Loomise, muutmise ja viimase kasutamise kuupäev
- Looja või omanik

Faili tüübi määramiseks on mitmeid erinevaid võimalusi: Microsofti operatsioonisüsteemid lisavad failile faililaiendi, Linux kirjeldab faili tüübi ära numbri abil ning näiteks MacOS lisab faili programmi nime, millega fail loodi jne. Faili tüüp omakorda vihjab enamus juhtudel sellele, milliseid andmeid failis hoitakse ja mis rakendus seda failis olevat infot tõlgendada oskab

Failide asukohta kirjeldatakse tavaliselt kahest vaatenurgast: kus fail füüsiliselt andmekandjal asub ning milline on faili asukoht failisüsteemi mõistes. Faili asukohta failisüsteemis kasutaja jaoks kirjeldavad enamasti kataloogid (kaustad), kuid viis, kuidas failisüsteem katalooge kirjeldab võib olla väga erinev: näiteks Microsofti failisüsteemides on olemas eraldi kataloogi objekt, UNIX'i laadsed operatsioonisüsteemid mõistavad katalooge kui teatavat kindlat liiki faile.

Ajalooliselt on kasutusel olnud ka failisüsteeme kus katalooge ei kasutata, selliseid failisüsteeme nimetatakse ühetasandilisteks failisüsteemideks (*flat filesystem*). Nende failisüteemide peamine puudus on see, et igal failil sellises failisüsteemis peab olema kordumatu nimi. Enamik tänapäeval kasutatavad failisüsteemid on hierarhilised failisüsteemid (*Hierarchical File System*), mis tähendab seda, et iga fail sellises failisüsteemis kuulub mõnda kataloogi. Kõik kataloogid kuuluvad omakorda ise mõnda kataloogi väljaarvatud üks kataloog, mida nimetatakse juurkataloogiks (*root directory*).



Ligipääsu- ja kasutusõigused ning omanik või looja ei pruugi olla kõigis failisüsteemides faili omaduste hulgas (näiteks FAT failisüsteem ei kirjelda õiguseid failide suhtes ning faili loojat või omaniku ei kirjeldata ka). Sellisel juhul on tegemist ebaturvalise failisüsteemiga ning selliseid failisüsteeme ei soovitata tänapäeval kasutada.

Logivad failisüsteemid (tuntud ka kui päevikut pidavad failisüsteemid, ik *journal*) on failisüsteemid, kus transaktsioonid (mingi hulga andmete muutmised) logitakse ehk siis märgitakse üles, mida ja kus muudeti või millega asendati ning alles seejärel tehakse see transaktsioon failisüsteemis reaalselt. Kui nüüd transaktsiooni teostamise ajal tekib süsteemi töös tõrge, siis süsteemialglaadimisel on võimalik poolikud transaktsioonid tagasi võtta ning teostamata muudatused teostada.

Päevik on faili, andmebaasi vms muudatuste kohta teavet sisaldav andmestruktuur. Päevik paikneb harilikult spetsiaalselt selleks reserveeritud piirkonnas. Failisüsteemid on enamasti väga suured andmestruktuurid ja nende värskendamine võtab palju aega. Kui keset värskendamist toimub süsteemi kokkujooksmine, siis tuleb failisüsteemi taastamiseks läbi käia kogu failisüsteem. Päeviku kasutamise mõte on selles, et muudatusi sisaldav fail salvestatakse kõigepealt päevikusse ja alles hiljem failisüsteemi. Kui seejuures peaks tekkima probleeme, on alati käepärast päevikusse salvestatud failikoopia ning piisab ainult selle faili uuesti salvestamisest. Taolist füüsilist päevikut kasutab näiteks Linux'i failisüsteem ext3. Teine päeviku variant on loogiline päevik, kuhu salvestatakse mitte andmeid endid, vaid neid kirjeldavaid metaandmeid. Loogilist päevikut kasutab näit. failisüsteem XFS vallaste.ee

Mõned tuntumad failisüsteemid:

FAT (*File Allocation Table*) on Microsofti operatsioonisüsteemides kasutatav failisüsteem, mida kaasaegsetes Micosofti operatsioonisüsteemides ei soovitata kasutada. Kuid kaheldamatult on tegemist ühe levinuma failisüsteemiga mobiilsetes

mäluseadmetes (USB-välkmälud, MP3 mängarid, flopid jne). Tegelikult on FAT failisüsteemil olemas tervelt kolm levinud versiooni: FAT12, FAT16 ja FAT32.
FAT12 oli algselt mõeldud kasutamiseks ainult floppidel, FAT12 kasutab andmeblokkide kirjeldamiseks 12 bitti, ehk kokku suudab adresseerida kuni 4096 erinevat andmeblokki, iga andmeblokk mahutab 512 ehk siis loogilise ketta maksimaalne suurus on FAT12 failisüsteemis 2048 kB, mis teeb selle failisüsteemi praktiliselt kasutamatuks kõvaketastel.
FAT16 failisüsteemi maksimaalne loogilise ketta suurus on kuni 2GB, mis oli kunagi täiesti piisav, kuid mida tänapäeval on kõvaketaste haldamiseks selgelt liiga vähe. Kuid FAT16 failisüsteemi suudab lugeda ja sinna kirjutada pea iga kaasaegne operatsioonisüsteem.
FAT32 failisüsteemi maksimaalne loogilise ketta suurus on kuni 8TB, kuid samas on piiratud maksimaalne faili suurus failisüsteemis 4GB suurusega, mis kaasaegsete multimeediumrakenduste kasutamisel saab kiirelt vägagi häirivaks.
FAT failisüsteemi suurim puudus on failiõiguste puudumine ning suurim tugevus see, et enamik operatsioonisüsteeme on võimelised seda failisüsteemi kasutama!

NTFS (*New Technology File System*) on tänapäeva Microsofti operatsioonisüsteemides enim kasutatav failisüsteem. NTFS failisüsteemis on failidel olemas kasutajaõigused, võimalus failisüsteemi tasemel failide pakkimiseks või krüpteerimiseks, maksimaalne loogilise ketta ja faili suurus on 16EB (16 700 000 TB). NTFS on logiv failisüsteem.
NTFS sisaldab mitmeid kaasaegsele failisüsteemile iseloomulike omadusi, näiteks võimaldab NTFS ühendada (*mount*) uue loogilise ketta olemasoleva loogilise ketta kataloogi külge, võimaldab failisüsteemi indekseerida jne.
NTFS suurim puudus on, et ainult Microsofti kaasaegsed (NT-seeria) operatsioonisüsteemid suudavad NTFS failisüsteemi vormindatud loogilisi kettaid probleemivabalt kasutada.

Ext2 on Linux operatsioonisüsteemides kasutatav failisüsteem. Ext2 failisüsteemis on failidel olemas failiõigused ning maksimaalne lubatud failisuurus on 2 TB ning loogilise ketta maht 32 TB. Failisüsteemi puuduseks on nõrk tõrketaluvus: kuna tegemist ei ole logiva failisüsteemiga, siis süsteemi tõrke korral võib juhtuda et failisüsteem (või selle osa) rikneb.
Ext2 ning ka paljude teiste Linuxi operatsioonisüsteemis kasutatavate failisüsteemide omapära seisneb selles, et kataloogid on teostatud nagu failid, mis viitavad teistele failidele.

Ext3 on ext2 failisüsteemi edasiarendus, lisatud on logiva failisüsteemi tugi. Enamik eeliseid ja puuduseid on samad, mis ext2 failisüsteemil, kuid parandatud on just tõrketaluvust. Enamik tänapäeva Linuxi distributsioone kasutab just etx3 failisüsteemi.

RaiserFS on samuti Linux operatsioonisüsteemides kasutatav logiv failisüsteem. RaiserFS failisüsteemis on failidel olemas failiõigused ning maksimaalne lubatud failisuurus on 8 TB ning loogilise ketta maht 32 TB. Üldiselt loetakse RaiserFS failisüsteemi kiiremaks kui ext2 ja ext3 failisüsteeme, kuid samas on RaiserFS failisüsteemil olnud üsna suuri probleeme stabiilsusega.

HFS+ (tuntud ka kui Mac OS Extended)on tänapäeva MacOS operatsioonisüsteemides kasutatav failisüsteem. HFS+ failisüsteemis on failidel olemas failiõigused ning maksimaalne lubatud failisuurus on 8 EB ning loogilise ketta maht 32 EB. Failisüsteemi omapära seisneb selles, et failidega seotud meta-andmeid, failitabelit ja kataloogifaili sisu hoitakse B* puus. Sama failisüsteemi kasutab ka Apple Corp toodetud meediamängarid iPod (samas võivad kasutada ka FAT32 failisüsteemi).

Meta-andmed ehk "andmed andmete kohta" on andmete digitaalne kirjeldus, mis vastab täpselt defineeritud mallile või ka selliste kirjelduste kogumik. Sarnaneb raamatukogu kataloogisüsteemile, kust sageli leiab lisaks andmetele raamatute nimetuste ja nende autorite kohta ka lühikese sisututvustuse
vallaste.ee

C.2.4 Turvalisus ja kaitse

Eesmärgid
- Teada arvutisüsteemi kaitse ja turvatoimingute vajadust
- Kirjeldada operatsioonisüsteemides realiseeritud kaitsemehhanisme
- Mõista erinevust identifitseerimise ja autentimise vahel
- Teada taaste ja varunduse vajadust
- Kirjeldada mõisteid "tagauks", „trooja hobune" ja arvutiviirus

Turvalisus on tänapäeva arvutisüsteemides väga oluline ning tarkvaratootjad pööravad sellele järjest suuremat tähelepanu.

Turvalisuse kolm põhikriteeriumit:
- Käideldavus (*availability*)
  - Andmed peavad olema takistusteta kättesaadavad volitatud kasutajatele
  - Andmed peavad olema õiged
- Terviklikkus (*integrity*)
  - Andmeid ei tohi volitamatult muuta
  - Andmed peavad olema pärit õigest allikast
- Konfidentsiaalsus (*confidentiality*)
  - Andmed tohivad olla kättesaadavad ainult volitatud kasutajatele

Täiendavateks kriteeriumiteks loetakse:
- Töökindlus *(reliability)*
- Autentsus *(authenticity)*
- Jälitatavus (*accountability)*

Kui mõni põhikriteeriumitest ei ole täidetud, siis võib öelda, et tegemist on ebaturvalise süsteemiga. Kaasaegsetesse operatsioonisüsteemidesse on sisseehitatud mitmeid kaitsemehhanisme turvakriteeriumite täitmiseks: kasutajad autenditakse, kasutajale rakenduvad teatud piirangud (tal on õigus kasutada ainult kindlaid ressursse). Mitmed rakendused kontrollivad andmeallikaid ning väljastavad hoiatuse, kui andmeallikas ei ole õige jne.

2.4.1 Turvaohud

Turvaohte on väga erinevaid. Laia kõlapinda on leidnud tihti just arvutivõrgu abil toimuvad ründed arvutisüsteemidele ning arvutiviirused, kuid tihti kipuvad kasutajad unustama arvuti füüsilist turvalisust.

Füüsiline risk: arvutisüsteem võidakse varastada, hävitada; arvutisüsteem muutub kasutatamatuks riistvara rikke või elektrikatkestuse tõttu jne.

Iga arvuti, mille korral on võimalik volitamata füüsiline ligipääs on potentsiaalne turvaoht. Pea iga kaasaegse operatsioonisüsteemi turvavahendid on vaikimisi seades turvamata või lihtsalt murtavad füüsilise ligipääsu korral. Seega peab olema arvutisüsteemile füüsiline ligipääs ainult selleks volitatud isikutel. Serveriruum peab olema eriti range kontrolli all, kindlasti tulekustutussüsteemiga varustatud, ventileeritav ja lukustatav ruum.

Riistvara abil on võimalik tagada ka teatud tasemeni tõrkekindlus riistvararikke vastu. Selleks dubleeritakse võimalusel vastavad süsteemid. Tuntum selline lahendus on kindlasti RAID (*redundant array of inexpensive disks*), mis võimaldab muuhulgas mitme kõvaketta kasutamisel tõsta oluliselt tõrkekindlust. Vajadusel säilitab arvutisüsteem seeläbi töövõime ja andmed ka kõvaketta tõrke korral. Kaasaegsetel tõrkekindlatel arvutisüsteemidel on tihti ka dubleeritud protsessorid, mälu, toiteplokid jne.

Kõige lihtsam turvarisk on kaheldamatult elektrikatkestus, selle vältimiseks on lihtsamatel juhtudel kasutusel katkematutoiteallikas ehk UPS (*Uninterruptible Power Supply*) . UPS on oma olemuselt suure mahutavusega akumulaator, mis võimaldab arvutisüsteemi töö jätkumise ka elektrikatkestuse korral. Kõige levinumad on lokaalsed UPS seadmed, mis ühendatakse mõne konkreetse arvutisüsteemi külge ja mis tagavad sellisel juhul selle konkreetse arvutisüsteemi töö. On olemas ja kasutusel ka suuremad UPS seadmed, milliste abil on võimalik tagada ka mõne ruumi või suisa hoone tõrkekindlus elektrikatkestuse korral. Eriti missioonikriitilistel asutustel on peale UPS seadme kasutusel ka oma elektrigeneraatorid, mis käivituvad elektrikatkestuse tekkimisel.

Kasutajapoolne risk: kasutaja, kellel on liiga suured volitused on samuti turvarisk, seda eriti juhul kui arvutisüsteem ei toeta kasutajaõiguseid või on muu moel kasutajapoolse rünnaku või väärkasutamise vastu turvamata.

Vanemates operatsioonisüsteemides puudusid tihti võimalused kasutajate eristamiseks või nende volituste piiramiseks. Sellised operatsioonisüsteemid on tänases mõttes täiesti ebaturvalised. Kõik kaasaegsed operatsioonisüsteemid sisaldavad vahendeid kasutajate eristamiseks, tuvastamiseks, autentimiseks ning nende õiguste piiramiseks. Uusimates operatsioonisüsteemides ei lubata kasutajaid tihti ilma tungiva vajaduseta operatsioonisüsteemi kasutada süsteemihalduri õigustes. Sellisel juhul antakse kasutajale süsteemihalduri õigused vaid ainult nende tegevuste teostamiseks, kus süsteemihalduri volitused on tõesti vajalikud (sageli nõutakse sellisel juhul ka korduvat autentimist). Nii püütakse välistada ka olukorda, kus kasutaja poolt tahtlikult või tahtmatult käivitatud pahavara ei saa teostada tegevusi (süsteemi kahjustamist) süsteemihalduri volituste abil.

Ühed kaasaja suurimad turvaohud moodustavad kindlasti arvutiviirused, pahavara ja võrguründed.

Võrguründeid selles moodulis ei käsitleta.

Turvariskide maandamiseks kasutatakse varukoopiad (*backup*) ehk varundamist. Kui mõni risk realiseerub on võimalik tagavarakoopiate olemasolul andmed taastada. Kui süsteemi töövõime on kriitilise tähtsusega, siis peavad tagavarakoopiad olema kättesaadavad koheselt ehk sellisel juhul peab varundus olema riistvaraliselt teostatud nii, et andmed on tagavarakoopiatelt kiirelt (soovitavalt reaalajas) taastatavad. Enamik kaasaegseid operatsioonisüsteeme sisaldavad endas varundus- ja taastusvahendeid.

2.4.2 Arvutiviirused ja pahavara

Arvutiviirused ja pahavara on üheselt kirjeldatav kui kahjulik tarkvara. Sellise tarkvara eesmärk on kahjustada arvutisüsteemi, kasutada arvutisüsteemi ressursse mitte selleks mõeldud tegevusteks ning segada kasutajat.

Levinumad pahavaraliigid on: arvutiviirus, uss, rootkit, nuhkvara, reklaamvara ja

trooja hobune.

Kõige tuntum pahavara on kindlasti arvutiviirus. Siinkohal tuleb mainida, et mõningate käsitluste kohaselt on arvutiviirust käsitletud ka kui pahavara üldistavat mõistet.

Arvutiviirus on programmikood, mis on kirjutatud selge sihiga, et see end ise paljundaks. Viirus üritab levida arvutist arvutisse, kinnitades end peremeesprogrammi külge. See võib rikkuda tarkvara, andmeid ja isegi riistvara.

Arvutiviirus on kopeerib end mõne teise programmi koodi ning kui see programm käivitatakse, siis käivitub kõigepealt viiruse kood ning alles seejärel programm ise. Nii saab arvutiviirus ligipääsu arvutisüsteemi ressurssidele ning hakkab seejärel otsima uusi „puhtaid" faile kuhu oma kood kopeerida. Seepärast öeldaksegi tihti, et arvutiviirus on programm, mis suudab ise levida. Tegelikult arvutiviirus päris iseseisvalt ju ei levi, ta vajab programmifaile (või teatud juhtudel mõnd teist liiki faile), kuhu oma koodi kopeerida. Faile, kuhu viiruse programmikood on kopeeritud nimetatakse peremeesfailideks (*host*).

Olemas on ka arvutiviiruseid, mis nakatavad näiteks Microsoft Wordi dokumendifaile (makroviirused) jne. Sellisel juhul hakkab viirus levima arvutisüsteemis juhul kui nakatunud fail mõne rakendusega avada. Tänapäeval on sellist liiki viiruseid toodetud pea kõikide levinumate rakenduste failidele ning viirus võib levida ka näiteks piltide, e-raamatute ja muude taoliste rakenduste failide abil.

Uss (*worm*), nagu ka viirus, on konstrueeritud ennast ühest arvutist teise kopeerima, kuid ta teeb seda automaatselt, kasutades arvuti funktsioone, mis on mõeldud failide või andmete transpordiks. Kui teie arvutisüsteemis on uss, võib ta iseseisvalt edasi liikuda. Usside võime massiliselt paljuneda kujutab endast suurt ohtu. Näiteks võib uss saata enese koopiaid kõigile teie aadressiraamatus leiduvatele e-posti aadressidele, teha sama igas järgmises arvutis, kuhu tal õnnestub levida, ning põhjustada doominoefektina väga palju võrguliiklust, mis võib aeglustada ettevõttevõrke ja internetti. Uued ussid levivad väga kiiresti ja ummistavad võrke, pannes teid (ja kõiki teisi) ootama internetis veebilehekülgede kuvamist võib-olla kaks korda kauem.
http://www.microsoft.com/eesti/security/home/antivirus/virus101.mspx

Nagu öeldud on ussi ja arvutiviiruste suurim erinevus just levimise tehnoloogias: kui arvutiviirus levib kasutades teisi faile, siis uss levib kasutades võrguteenuseid. Nii ussi kui ka viiruse tegelik eesmärk ei ole kindlasti vaid enese levitamine, vaid mingil kindlal juhul hakkab viirus või uss süsteem kahjustama (näiteks kustutab faile, edastab infot, segab kasutajat jne).

Trooja hobune on arvutiprogramm, mis näib olevat kasulik, kuid tegelikult teeb see hoopis arvutisüsteemile kahju.

Nii nagu müütiline trooja hobune nägi välja nagu kingitus, kuid sisaldas Kreeka sõdureid, kes Trooja linna vallutasid, on tänapäeva trooja hobused arvutiprogrammid, mis näivad kasulikud, kuid tegelikult ohustavad teie turvalisust ja võivad teha palju hävitustööd. Näiteks on üks

trooja hobune levinud meilisõnumina koos manusfailidega, näiliselt Microsofti turvavärskendustega; tegelikkuses osutusid need viirusteks, mis üritasid välja lülitada viirusetõrje- ja tulemüüritarkvara.

Rootkit on pahavara, mis kasutajanime ja parooli kasutamata hõivab juurkasutaja õigused. Rootkit kontrollib reeglina kõiki operatsioonisüsteemi põhifunktsioone ning seepärast on sellise pahavara leidmine ja hävitamine arvutisüsteemis eriti keeruline.

Rootkite on olemas mitmeid erinevaid liike, kõige kahjulikumad neist käivituvad enne operatsioonisüsteemi, võtavad kontrolli riistvara üle ning seejärel käivitavad operatsioonisüsteemi. Sellistel rootkitidel on kontroll kõigi operatsioonisüsteemi funktsioonide üle.
Rootkitide levitamiseks on väga erinevaid võimalusi: tihti levib rootkit sarnaselt arvutiviirusega, kuid samas on teada ka juhuseid, kus rootkit on levinud ka legaalsete muusikaplaatidega ning rootkiti valmistajateks on olnud väga tuntud ja lugupeetud ettevõtted. Kuna rootkit kontrollib enamasti kõiki operatsiooni põhifunktsioone, siis suudab see tarkvara väga hästi end varjata ning väga tihti võib juhtuda, et pärast rootkiti eemaldamist vajab operatsioonisüsteem täieliku väljavahetamist.

Tagauks on operatsioonisüsteemi või mõne rakendusprogrammi valmistaja poolt teadlikult tehtud võimalus arvutisüsteemi ressursside kasutamiseks ilma vastavaid volitusi omamata.
Samuti võib näiteks trooja hobune paigalda arvutisse spetsiaalse tagaukse, mille abil volitusteta kasutaja saab arvutisüsteemi kasutada.

Tagauksed olid ajalooliselt programmeerijate poolt endale kirjutatud lisavõimalused, seda ajal, kui turvalisust ei peetud veel oluliseks. Sellisel juhul oli rakenduse valmistajal (tugiteenuse pakkujal) võimalik arvutisüsteemi paigaldada uuendusi ning teha vajalike seadistusi ilma, et teataks näiteks vajaliku kasutaja nime ja parooli. Kaasajal selliseid lahendusi enam ei kasutata, sest sellisel juhul võib tagaust kasutada arvutisüsteemi kasutaja teadmata ning see ei ole lubatud.

2.4.3 Identifitseerimise ja autentimine

Kaasajal on kasutuses peamiselt mitme kasutajaga (*multi-user*) operatsioonisüsteemid. See ei tähenda mitte ainult seda, et operatsioonisüsteemis võib olla mitu kasutajat, vaid iga protsess sellises operatsioonisüsteemis töötab mingi kasutaja õigustes. Lisaks tavakasutajale on reeglina operatsioonisüsteemis olemas veel ka süsteemsed kasutajad, kelle õigustes töötavad süsteemi tööks vajalikud protsessid.

Kui me räägime nüüd tavakasutajatest (ehk inimestest, kes arvutisüsteemiga töötavad), siis sellised kasutajad tuleb tuvastada.

Kasutaja tuvastamine koosneb identifitseerimisest ja autentimisest

Identifitseerimise all mõistetakse reeglina kasutajaga seotud kasutajanime või mõne muu sarnase info (nimi, isikukood, id number jne) järgi kasutaja identiteedi leidmist (aga mitte veel tõestamine). Identifitseerida on võimalik end ka mõne füüsilise seadme abil (kiipkaart, USB seade jne).

Identifitseerimise käigus saadakse teada, kes kasutaja väidab end olevat

Kui on teada, kes kasutaja väidab end olevat, siis seejärel tuleb kuidagi kontrollida, kas kasutaja tegelikult on ka see, kes ta väidab end olevat. Sellist tegevust nimetatakse autentimiseks.

Autentimiseks nimetatakse väidetava identiteedi tõesuse kontrollimist

Sellise kontrolli teostamiseks on kõige tavapärasemal juhul kasutusel kasutajaga seotud salasõna (*password*). Kui kasutajanimi ja selle kasutajanimega seotud salasõna langevad kokku, siis loetakse kasutaja tuvastatuks. Ka autentimisel on kasutusel mitmeid erinevaid võimalusi: kaasajal on hakanud levima biomeetriliste (näpujäljed, silma võrkkest jne) andmete kontrollimine vastavate seadmete abil.