Ettevõtte varundus luubi all: hoia oma digivara!

16. august 2016 - 0:57 Autor: Ivar Pikker

Varundus on keeruline teema - see pole pelgalt andmete kopeerimine kuhugi ja nende sealt võtmine, kui midagi juhtub. Kuidas kopeerida, millal ja mida, küsimusi on palju. Siin on väike ülevaade, mis on Enterprise klassi varundus ja kuidas seda teha.

Image põhine varundus

Varundust võib jaotada kahte olulisemasse gruppi - operatsioonisüsteemide varunduseks ja andmete varunduseks. Kummagi jaoks on täiesti erinevate brändide tooted ning pingutada nende varunduste surumisega ühe brändi toote alla annab halva tulemuse.

Kõigepealt aga räägime opsüsteemi varundusest.

Peamine vahe seisneb varundustehnikas. Opsüsteemi varunduseks sobib hästi sektoripõhine varundus ja andmetele failipõhine varundus. Asi pole ainult selles, et operatsioonisüsteem võib kasutada ka failisüsteemile kättesaamatuid sektoreid, vaid kaks olulist lisategurit on veel usaldusväärsus ja kiirus. Varundustarkvara VSS toega agent toimib küll *low-level*il (opsüsteemi enda *level*), kuid alati on võimalus, et opsüsteemi on installitud mingi tarkvara, mis arvab ennast veel tähtsama olevat ning võtab juhtimise üle veel enne VSS-i. Näitena võiks tuua igasugu opsüsteemi *rollback* tarkvarad.

Teine tegur on kiirus. Sektoripõhine varundus on kiirem sellepärast, et loeb failisüsteemi madalamal tasandil kui failipõhine varundus. Eelis saadakse ka siis, kui tegemist on paljude väikeste failidega, mis opsüsteemi puhul ongi tavaline. Peamine eelistus aga ei tulene mitte turvalisuse või kiiruse teguritest, vaid loogikast. Varunduse loogika peab olema alati lähedal sellele struktuurile, mida varundatakse. Opsüsteem on üks tervik, seega mõistlik on kasutada *low-level* tasandil failisüsteemi lugemist. Andmed (ükskõik, kas andmebaas või failiserver) on üksteisest sõltumatu struktuur, info on personaalne. Kui opsüsteemi hetkeseisul tehtud varundus võib suure tõenäosusega töötada ilma tõrgeteta, siis poolikud andmed enamasti kajastavad infot valesti. Andmetest ei saa teha hetke ajaseisu tõmmist vaid andmed peavad olema lõpetatud mingi loetava seisuni.

OK, nüüd lähemalt sektoripõhise varunduse juurde, mida tihti nimetatakse ka *image*-põhiseks varunduseks. Siin pole mängumaa väga lai ja ette rutates võiks kohe öelda, et liider on Acronis (täpsemalt Acronis Backup Advanced, mitte see Home). Acronis on aastate jooksul teinud märkimisväärset karjääri. Peamisteks konkurentideks nimetaks R1Soft-i ja StorageCraft-i toodet ShadowProtect-i. Ja ongi kõik. Väga paljud brändid lisavad oma tootevalikus sektoripõhise variandi varundusest, kuid on astunud hilisemale rongile, kui mitte öelda otse, et need on rongilt lausa maha jäänud ja ei tea, millega simitsi seisavad.

R1Soft on erandlikult unikaalse lahendusega. Kontseptsioon kujutab tervet varunduse *destination storage´*it kui monoliitset andmebaasi. Eelis on selles, et rotatsioon (siis varunduspunktide-*snapshot*ide konsolideerimine) on väga kiire, Windows Journali kasutus on arenenum kui Acronisel. Windows Journal tähendab seda, et opsüsteem peab logi kõigi muudatuste kohta kettal. See Journal võib olla ka varundustarkvara enda logi. Kusjuures oluline on ära tunda, millal see logi või Journal muutub kasutuskõlbmatuks, kuna vale Journali alusel varundus on ohtlik. Journal aga muutub kasutuskõlbmatuks kohe niipea, kui kettale tehti mingi muudatus enne Journali pidava driveri laadimist.

Tõsisteks R1Soft puudusteks aga on väga ebamugav BareMetalRestore (ehk siis opsüsteemi taastus) ja aeglane varundus juhul, kui Windows Journal pole kättesaadav. Muidugi lisaks ka deduplitseerimine on saadaval nendest kolmest asinult Acronisel. StorageCraft on vana tehnoloogiasaurus, mille jaoks Acronis oli pigem õpipoiss. ShadowProtect omab mõningaid huvitavaid funktsionaalseid eeliseid Acronise ees, kuid pole siiski päris Enterprise klassi toode. Mis puutub *image*-põhist varundust, siis Acronis oli algul üldse imiteerija rollis. Tehnoloogia eestvedajad olid 90ndatel hoopis PowerQuest, Storagecraft ja Symantec, kui jutt on *image*-põhisest varundusest. Täna aga nendest ainult Acronis ja R1Soft on jõudnud arvestatava *image*-põhise Enterprise leveli varundustarkvara hulka. See tähendab siis eeskätt keskhaldust.

Failipõhine varundus

Nüüd aga failipõhisest varundusest, mis on tunduvalt mahukam teema ja natuke teine maailm. Failipõhine Enterprise varunduse maailm on olnud aastakümneid suurte kontsernide pärusmaa ja on seda siiani. Paljud tarkvara tehnoloogiaga seotud funktsionaalsed nimetused on lausa klassika ja selle tõttu kaasa saanud ka stagnatsiooni tunnused. Viimasel ajal on aga ka siin brande mis harjumuspärase pildi segi löövad.

Suurim tegija on olnud Veritas aastast 1989 Californiast. Muidugi on varundusega tegeletud juba sama kaua kui on olnud arvutid, kuid Veritase puhul räägime põlvkonnavahetusest. Siiski 5 aastat varem alustas samuti Californiast üks teine võtmetähtsusega firma, algselt Mac´idele mõeldud Dantz (Vahepeal reisinud EMC ja Roxio alla, nüüd Retrospecti enda tiimi all tagasi), millest peagi räägime lähemalt.

Enterprise klassi tuntumad varundusbrandid ja nende tooted on Veritas NetBackup, Veritas Backup Exec, Dantz Retrospect, IBM Tivoli Storage Manager, EMC (varem Legato) Networker, EMC Avamar, Arcserve, CommVault, HP Data Protector, Acronis Backup Advanced. Lisaks veel suhteliselt uus ja kiirelt populaarsust kogunud virtuaalmasinate varundusbrand Veeam, aga see on eraldi teema ja sellest ka juba pisut hiljem.

Üks mantraga võrreldav sõna on *INCREMENTAL*, mis on igasuguse Enterprise varundustarkvara lahutamatu osa. Varundatakse ainult muudatusi. Täisvarundust ei jõua keegi enam tänapäeval igapäevaselt teha. See võtab liiga kaua aega, mõnikord päevi suurte andmemassiivide puhul, koormab ebaratsionaalselt nii serverite kui klientide masinate ressursse ning raiskab kallist varundusruumi. Esimene küsimus kohe on siis see, et mida pidada muudatuseks, et seda varundada. Erinevad brändid peavad selle all silmas väga erinevaid asju. Klassikalise faili muudatuse kriteeriumiks kasutati kaua arhiivi lippu faili atribuutides. Opsüsteemil on reeglina kohustus kustutada arhiivi likupe peale igat muudatust failis. Tänaseks on selgunud, et sellel meetodil on mitmeid puudusi. Siiski mitte alati ei kustutata arhiivi lipuke peale faili muudatust, näiteks mingi süsteemse tarkvara sekkumise puhul. Samuti näiteks kopeerimise puhul küll seatakse püsti arhiivi lipuke, kuid muudatust failis siiski polnud. Teiseks jääb arhiivi lipu meetodit kasutatava tarkvara puhul jäigalt seotuks varundatud sessioon ja lähtekettal olev fail. Kui näiteks kustutada varundatud sessioon, näiteks mingi pooliku varunduse vea tõttu, siis edasised *incremental* varundused pole enam võimalikud, kuna algse faili arhiivilipud on juba kustutatud ja enam pole võimalik kindlaks teha, milline fail oli muudatusega.

Tunduvalt kindlam meetod on pidada andmebaasis-kataloogis (kataloogiks nimetatakse andmebaasi, mis kajastab kõiki varundustoiminguid ilma, et varundatud *destination-storage-media* oleks kättesaadav) varundatud faili atribuutide andmeid ja järgmise *incremental* varunduse puhul võrrelda neid kataloogi andmeid reaalselt kettal oleva faili atribuutide andmetega. Windowsi operatsioonisüsteemi puhul nendeks usaldusväärseteks atribuutideks oleksid faili nimi koos aadressiga, faili suurus, faili loomise aeg ja faili muutmise aeg.

Kusjuures väga oluline on, et igasugune muutus tähendaks muudetud faili staatust. Ideaalseim varundustarkvara selles suhtes on Dantz Retrospect, mis täpselt just neid nelja parameetrit kontrollibki, arhiivi lipp aga ei mängi üldse tähtsust.

Veel ohtlikum arhiivi lipu kontrollist aga on faili muudatuse aja kontrolli meetod nii, et faili muudatuseks loetakse ainult värskeimat aega. Siin tekib kohe oht, et kui varundusest fail on taastatud, siis ta muutmise kuupäev on palju varasem ning seega langeb ta varundusest välja, mis aga on viga, sest ka kõik taastatud failid tähendavad muudetud faili võrreldes eelmise versiooniga. Kõige ohtlikum ja ebakindlam meetod on kasutusel Veritase (vahepeal Symanteci all olnud) NetBackup´il. NetBackup võrdleb algse faili muutmise aega, mitte kataloogi andmebaasi andmetega selle faili kohta, vaid viimase varunduse ajaga. Lisaks sellele, et varasema ajatempliga failid jäävad varundamata, sõltub selline võrdlus tugevalt ka serveri-kliendi masinate kellaaegade nihkest. Kui serveri aeg on varasem kliendi omast, varundatakse pidevalt samu identseid faile. Kui serveri aeg on hilisem kliendi omast, jäävad viimatimuudetud failid varundamata. Veritas on selle tarbeks loonud veel ühe väga ümbertnurga abivahendi nimega "*time overlap*", aja seade kliendi seadetes, mis siis liidab algse faili muutmise ajale juurde selle seadetes määratud numbri, et kindlalt fail ikka varundada, kuigi sellega kaasneb osade failide topeltvarundus.

Õnneks saab NetBackupi ka alternatiivselt seadistada, et kasutaks Journali või *accelereation*i, mis võtab veel lisaks parameetreid appi võrdlusse. Aga ka see on ikkagi selline halva tehnoloogia ravi. Perfektne lahendus on Dantz Retrospectil. Samuti on kasulikum, kui varundustarkvara arhiivi lipu üldse rahule jätaks ja seda peale varundust ei muudaks.

Järgmine teema on, kuidas vähendada lõputult kasvavat *incremental*ite hulka, ehk siis *consolidation*, *merge*, *grooming* või *synthetic backup*, vastavalt kuidas mingi bränd selle on lahendanud või seda nimetab. Siin on sisuliselt kaks erinevat lähenemist ja tehnikat. Üks on pidev viimase täisvarunduse liitmine viimase *incremental*iga, saades nende asemele uue täisvarunduse. Seda meetodit kasutab Dantz

Retrospect, Acronis, Ahsay. Teine meetod on sünteetiline (Syntethic) varundus, kus viimasest täisvarundusest ja kõigile järgnevatest *incremental*itest luuakse uus ftäisvarundus. See meetod on laiemalt kasutuses.

Lisaks Veritas NetBackupil on võimalik veel moodustada uus *incremental* ainult kõigist varasematest *incremental*itest. Kõige paindlikum lahendus on Dantz Retrospectil. Grooming konsolideerib-liidab kokku kõik varasemad (seadetes saab määrata, mitu viimast *incrementali-fulli* alles jätta). Valikuliselt saab varundussessioonidele panna peale blokeeringu groomingu jaoks, mis tähendab, et antud sessioon jääb alles ja groomingust välja.

Nüüd on väga oluline, kas varundustarkvara suudab ka iga varundussessiooniga salvestada momendi failipuu tõmmise-hetkeseisu. See on vajalik selleks, et taastusel mitte taastada juba kettalt kustutatud faile. Näiteks kõigepealt toimus *full backup*, siis kustutati kettalt mingi fail ja siis tehti uus *incremental* varundus. Ilma selle infota kataloogis taastaks varundustarkvara viimase *incrementali* järgi ka selle kustutatud faili. Vastava info olemasolul aga varundustarkvara näeb, et taastatava sessiooni varunduse ajal seda faili enam polnud ning seda ei taasta, kuigi varasemas sessioonis on ta varundatud. Reeglina kõik Enterprise klassi varundustarkvara oskab seda infot salvestada.

Veritas NetBackup-il tuleb see eraldi aktiveerida "TrueImageInfo" nime alt. Üldiselt tarkvara, mis sellist infot ei käitle, nimetatakse versiooning-varundustarkvaraks ja sellel pole palju ühist Enterprise klassi kuuluva tarkvaraga.

Edasi on oluline, kuidas varundustarkvara oskab andmeid taastada. Ideaalne varundustarkvara oskab taastada nii *point-in-time snapshot*i alusel (siis täpselt sama seis, mis oli varunduse ajal kettal), kui ka ainult konkreetse sessiooni alusel (ainult failid, mis varundussessioonil lisati *incremental*iga). Acronisel (millel on ka suht nigel failipõhise varunduse liides) sessioonipõhist ülevaadet pole. Enamus siiski seda võimaldavad, kuigi mõnedel on informatsiooni esitamisega probleeme.

Näiteks Netbackup küll suudab taastada, kuid faile kuvada näitamisaknas ei suuda, ainult kaustu. Arcserve kuvab ainult sessiooni kuvamise aknas, kus peaks kuvama kõik sessioonil lisatud failid, üksiti ka kõikide kaustade puud, mis eksisteeris allikas varundussessiooni ajal, mistõttu faile üles leida oli raskendatud. HP Data Protector aga kahjuks kuvab *point-in-time* aknas üldse kõiki faile, mis kunagi varundatud, kuigi taastab korrektse ketta seisu.

Järgmisena on ka oluline, kuidas just täpsemalt taastada on võimalik. Kõik lahendused peale Dantz Retrospecti ei võta kunagi arvesse neid faile, mis enne taastamist juba on kettal. See tähendab, et kui tahad täpset ketta seisu, mis oli *point-in-time* varunduse ajal, siis enne taastust pead kettalt käsitsi kõik failid kustutama, et mitte jätta alles ebaolulisi faile. Dantz Retrospect aga teeb selle töö ise ära *on-the-fly* ja peale taastamist jäävad täpselt need failid, mis sai taastatud varundusest, ilma millegi üleliigseta. Sellel on veel teinegi hea omadus - kiirus. Kui Dantz Retrospect näeb et fail on identne, siis ei pea ta hakkama seda taastama.

Edasi on tähtis, kuidas saab salvestusmediat jagada erinevate asukohtade vahel, või peab see asuma sama varunduspoliitika jaoks ühes kohas. Acronis, Ahsay, Arcserve ei võimalda sama varunduspoliitika raames jagada *storage-media* asukohta.

Teistel enamasti pole selliseid piiranguid. Näiteks saab täisvarunduse lasta kirjutada ühe ketta peale ja *incremental*id teise ketta peale. Backup Execil saab poliitikas konkreetselt määrata, kuhu läheb *full*, kuhu *incremental* ja kuhu *synthetic* varundus. Samas Backup Exec ei võimalda samale kettale rohkem kui ühe *storage-media* ning *incremental*eid ei saa teha *deduplication pool*ile. Dantz Retrospectil pole mingeid piiranguid, kuid policis seda ei saa määrata, vaid tuleb käsitsi panna ajutine "skip" märge mediale.

Tõhus tehnika - deduplitseerimine

Edasi on *storage* meedia ruumi kokkuhoiu tarbeks loodud pakkimisest veel tõhusam tehnika - deduplitseerimine. Seda on kahe sorti. Rohkem delta-copy nime kandev deduplication toimib ainult sama faili raames ja terve faili varunduse asemel varundab ainult faili muudetud blokid. Deduplicationi nime kandev tehnika on globaalne ja toimib terve varundusmeedia raames, salvestades kõigest ainult unikaalse fraktsiooni.

Deduplicationi eelis on väga väike salvestusmeedia kulu.

Deduplicationi puudus on väga tõsine koormus protsessorile ja mälukasutusele. Kui mingi fraktsioon hävib, siis kahjustab see paljusid faile. Deduplikatsiooni meediat ei saa jagada erinevate asukohtade vahel, peab asuma samas kohas.

Delta-copy deduplicationi eelis: Ei koorma serverit-klienti oluliselt. Kui fraktsioon hävib, siis kahjustab see ainult ühte faili. Meediat saab jagada vabalt erinevate asukohtade vahel.

Delta-copy deduplicationi puudus: Ei saavutada nii suurt kokkuhoidu kui globaalse deduplicationiga.

Enamasti on kõigil kasutuses ainult globaalne deduplication. Erandina ainult delta-copy meetodit kasutavad tarkvarast ainult Dantz Retrospect ja Ahsay.

Äkki peaks rääkima ka lindiseadmete toest?

Ei, tegelikult ei viitsi. See on tõesti teema, mis oleks vaja visata ajaloo prügikasti. Kõvakettal andmete säilitamine on palju turvalisem kui lintidel ja pealegi odavam ka.

Ja muidugi saab ka kettaid seadmest välja võtta. Olulisem on igasugu NAS seadmete tugi. Neile ju standardset kliendi agenti ei installi. Kasutusel on spetsiaalne "riistvara snapshot" tehnika, mis siis näiteks failiserverina kasutuses olevast kettaseadmest teeb raua toel

snapshoti ja saadab otse mediaserverile varunduseks. Siin ei tohi segamini ajada näiteks VSS snapshotidega, mis on windowsi opsüsteemi tarkvaraline komponent avatud failide, opsüsteemi sisemiste andmebaaside ja point-in-time seisu varunduseks. Rikkalik tugi igasugu erinevatele opsüsteemidele ja riistvara seadmetele on kõige suurem Veritase NetBackupil ja IBM Tivolil. Iseasi muidugi, et kas seda kraami ikka on kõike tarvis, kui nüüd üle mõelda veel.

Näiteks failiserveri varunduseks pole üldse hea kasutada mingit open-file toega snapshoti, sest kirjutuseks avatud fail pole consistentne varunduseks. Selle jaoks on olemas eraldi tarkvaralahendused mis kontrollivad ja peavad ülevaadet paraleelsest dokumenditöötlusest. Ja varundust võib vaadelda lihtsalt kui ühte klientidest.

Kui varundustarkvara võimaldab, aga ega tihti erilist mängumaad selles vallas varundustarkvara enda poolt eriti pakkuda pole, siis ideaalne oleks nii - kui fail on avatud ainult lugemisõigusega teistele (shared access), siis sellest varundada pole kasulik, kuna faili kirjutatakse ja peaks ootama kuni fail vabaks lastakse.

Kui on avatud excluseive access, siis nagunii ei saa lugeda ja tuleb samuti oodata. Kui fail vabaneb varunduseks, siis peaks ta lock-ima shared accessiga, nii et seda saab ainult lugeda, et ei segaks varundust (faili on muudetud varunduse ajal). VSS või muu open-file manager aga ei garanteeri nagunii consistencit, aga eks see ole adminni enda valik otsustada. Backup Execil näiteks saab lukustada faili varunduse ajal (ilma VSS-ita), kuigi ta avab ka ainult lugemisõigusega faile (sinna kirjutatakse momendil).

Dantz Retrospect läheneb asjale hoopis kompromissitult ja võimaldab enne varundust täielikult lahti ühendada failiserveri shared varunduse ajaks. Enne seda saadetakse cliendi kaudu veel sõnum kasutaja desktopile et näiteks 5 min pärast failiserver ühendatakse lahti. Töö ajal seda muidugi teha ei saa, kuigi öösel pole probleem.

Andmebaaside tugi on enamasti kõikidel, tihti eraldi tasustatavad. Standardsed on MSSQL ja Exchange moodulid. NetBackupil ja IBM Tivolil ka Oracle ja muu lisaks. Tihti aga ka sellest ei piisa, sest paljud tarkvaraprogrammid kasutavad mingeid enda lahendustel põhinevaid transaktsioone, isegi nimetamata neid andmebaasideks. Transaktsioon on siis mingid arvuti mälus kettale kirjutust ootavad andmed, mis peavad lõpuks kettal olema konsistentsed mingite muude andmetega teises kohas. Nende transaktsioonide eiramine enne varundust oleks sama mis töötaval arvutil restardi nupu vajutus - kunagi pole kindel kas midagi rikkus ära või mitte. Et tagada application quiescent consistency, tuleb enne ja pärast varundust kasutada varundustarkvara tuge käsurea käskluste või skripti käivitamisel. Vastav skrip peab viima siis tarkvara transaktsioonid konsistentsesse seisu. Näiteks enne sulgema programmi või service ja peale varundust taas käivitama.

Virtuaalmasinate varundus

Lõpuks võiks pisut tutvustada ka virtuaalmasinate varundust, mis on põhimõtteliselt teise ülesehitusega ja pisut keerukam. Virtuaalmasina varundus tähendab seda et virtuaalmasinas puudub varundustarkvara klient-agent ja jutt käib enamasti virtuaalmasina opsüsteemi varundusest. Edasi on küsimus selles et kuidas tagada application ja failisüsteemi consistency. Erinevad brandid lähenevad sellele väga erinevalt ning alati nad ei taha seda dokumenteerida. Samas on see väga oluline teada varundustarkvara valiku tegemisel. Kõige lihtsam on crash-consistency.

Hypervisor ootab hetke kuni kettale kirjutus rahuneb, flushib bufrid, teeb virtualketta snapshoti, varundab snapshoti ja siis kustutab snapshoti. See on sama kui lülida arvutil reset nuppu. Virtualmasina virtuaalketta varundus on mugavam ja kiirem kui opsüsteemi varundus agendi abil. Lisaks kasutatakse Changed Block Tracking (CBT) logi, mis pidevalt peab ülevaadet-logi virtuaalkettale kirjutuse kohta.

See on analoogne windowsi journalile, kuid töökindlam, kuna hyperviisor on madalam (raualähedasem) kui opsüsteemi draiver. Kõrgelt hinnatud Veeam virtuaalmasinate varundus on pisut intelligentsema tööpõhimõttega. Ka seal puudub agent, kuid hyperviisor teeb siiski virtuaalmasina abiutiliidi abil VSS snapshoti (nagu ka tavalise opsüsteemi varunduse puhul). Edasi teeb hypervisor virtuaalkettast snapshoti. Edasi rollib-back selle virtuaalketta snapshoti internal VSS snapshotiga identsesse seisu. Seega siis sisuliselt saadakse varunduseks kätte virtuaalmasina internal VSS snapshot, ainult enamus kopimist toimub hosti hypervisori tasemel virtuaalkettast. See on siis kuidas Veeam töötab. Kuid, ka see ei taga veel 100% virtuaalmasina consistencyt. Kui on tarvis täielikku 100% kindlust, siis pole muud varianti kui varundada virtuaalmasin koos tema snapshotil tehtud mälufailiga. Selleks kõlbab suht suvaline varundustarkvara, isegi VSS-i tuge pole tarvis. Soovitav oleks kui varundustarkvara toetaks deduplicationi või delta-copyt, sest virtuaalkettad on suured ja ainult osa infot selles muudetakse, pole mõtet iga kord kopida tervet virtuaalketta faili. Enne varundust tuleb skriptiga anda hypervisorile käsk teha virtuaalmasinast snapshot, varundada virtuaalkettad koos mälufailiga ja peale varundust skriptiga anda käsk snapshot kustutada. VSS-i pole tarvis, sest viimane virtuaalmasina virtualketta fail ei oma tähtsust. Kui hyperviisor teeb virtuaalmasinast snapshoti, siis muudab ta olemasoleva virtuaalketta faili read-onlyks ning edasise kirjutuse tarbeks loob uue faili. See uus fail on küll kirjutuseks avatud, kuid ei oma varunduse seisukohalt tähtsust. Olulised on just varasemad virtuaalkettad ja mälufail, mis moodustavadki virtuaalmasina snapshoti. Taastades snapshoti konfiguratsioonifailis on selle kohta info ja virtuaalmasina tagasi kerimine sellele snapshotile pole probleem. Consistencyga pole mingit probleemi, kuna virtuaalmasin jätkab täpselt sealt kus viimati pooleli jäi, täpselt isegi CPU õige käsu pealt. Ainus olukord kus seda meetodit kasutada ei saa on siis kui virtuaalmasin teeb mingeid transaktsioone üle võrgu. Virtuaalmasin küll jätkab õige koha pealt, kuid võrgu teises otsas asuv andmebaas võib asjast valesti aru saada, sest vahepeal on aeg edasi liikunud ja andmebaas enam ei oota neid vanu käske.

Head varundamist!

- Lahendused