WSOY

PC käyttäjän NIKSIKIRJA

PETTERI ★ JÄRVINEN

Saate digitaaliseen näköispainokseen

PC-käyttäjän niksikirja oli vuonna 1993 ilmestynyt kirja, johon kokosin niksejä, vinkkejä sekä kiinnostavia tiedonjyviä ja havaintoja PC-käytöstä.

Osa nikseistä muokkasi käyttöjärjestelmän apuohjelmien toimintaa puukottamalla niitä DEBUGilla suoraan kooditiedostoon. Moinen käytäntö olisi tänään ehdottoman no-no ja estetty myös teknisesti.

Kirja oli välityö kaikkien muiden projektien lomassa, mutta sitä oli hauska tehdä. Tiedossa oli, että seuraava vastaava niksikirja käsittelee Windows-maailmaa.

Alkuperäisessä paperiversiossa oli hakemisto, jonka tiedostot eivät enää olleet tallella ja siksi se puuttuu näköispainoksesta. Kirjan alkulehdellä oli motto *Vastaukset ovat olemassa. Pitää vain tietää mistä etsiä*.

Julkaisen kirjan vapaasti jaettavana pdf-versiona, jotta jälkipolville säilyy ajankuva PC/DOS-maailman alkuvuosista.

Espoossa lokakuussa 2020,

Petteri Järvinen

Sisällys

Esipuhe

Kun kirjoitin ensimmäistä mikrokirjaani vuonna 1985 minun oli pakko lähteä liikkeelle aivan alkeista. Koko ala oli uutta ja outoa, eikä lukijoilta voinut vaatia minkäänlaisia pohjatietoja.

Tänään tilanne on kokonaan toinen. Aloittelijoista on kasvanut innokas ja osaava tehokäyttäjien joukko, joka perusteet opittuaan haluaa käyttää mikroaan mahdollisimman tehokkaasti. Tehokäyttäjä tuhahtelee tavallisille DOS oppaille ja haluaa tietää asioita, joita perusoppaissa ei edes sivuta.

Juuri heitä — mikrotukihenkilöitä, tehokäyttäjiä tai asioista muuten vain kiinnostuneita — varten olen kirjoittanut tämän kirjan. Olen koonnut siihen tietämystä ja niksejä, joita olen lukenut, nähnyt ja itse keksinyt niiden reilun 10 vuoden aikana, joina olen käyttänyt DOSia. Olen ottanut mukaan myös joukon yleisiä ja mielenkiintoisia kysymyksiä, joita minulle usein esitetään niin kursseilla kuin kirjeitsekin. Toivon, että kirjan niksit auttavat tehostamaan mikronkäyttöä ja vastaavat jo ennalta kysymyksiin, jotka ovat vasta tulossa mieleen.

Vaikka mikron käytön asiat ovatkin periaatteessa helppoja ja yksinkertaisia, mistä tahansa asiasta saadaan vaikea kun siihen perehdytään riittävän syvällisesti. Tämä kirja tekee monesta helpoksi luullusta asiasta jälleen vaikean, mutta toivottavasti myös monesta aiemmin vaikeasta asiasta helpon.

Otan mielelläni vastaan myös uusia niksejä tai parannuksia niihin, jotka tässä kirjassa on esitetty. Parhaat tullaan lisäämään kirjan seuraaviin painoksiin lähettäjiensä nimellä varustettuna. Jaettu niksi on paras niksi!

Palautetta, niksejä ja kommentteja kirjan sisällöstä voi faksata numeroon (90) 466464 tai lähettää postilla osoitteeseen Tekniikantie 12, 01250 ESPOO.

Albufeira 2.10.92 - Tapiola 24.1.93
Petteri Järvinen

Yleistä

0

Älä kerro kenellekään, miten helppoa tietotekniikka on!

Maallikot ja aloittelevat mikronkäyttäjät kuvittelevat usein, että tietotekniikka on mystistä, vaikeata ja vaatii matemaattista lahjakkuutta. He ovat väärässä. Tietotekniikka on erittäin helppoa.

Helppous johtuu siitä, että ala on täysin keinotekoinen. Se ei perustu mihinkään löytöihin tai luonnonlakeihin, vaan kaikki on alusta lähtien ihmisten kehittämää. Keinotekoisen alkuperänsä vuoksi ilmiöiden toiminta tunnetaan täydellisesti ja toisin kuin monilla muilla aloilla, kaikkiin kysymyksiin on olemassa vastaukset. Tilanne on siis aivan toinen kuin esimerkiksi teologiassa, joka ei vielä tuhansien vuosien tutkimisen jälkeenkään ole päässyt selville edes kaikkein keskeisimmistä kysymyksistä.

Tietotekniikassa vastaukset ovat aina olemassa. Ainoa vaikeus on löytää oikea kirja tai henkilö, jolta kysyä.

Tästä huolimatta tietotekniikka saattaa tuntua vaikealta. Se johtuu yksityiskohtien runsaudesta ja siitä tavattomasta nopeudesta, jolla ala kehittyy. Pieniä yksityiskohtia on tavattoman paljon, joten kokonaiskuvan hahmottaminen ja syy-yhteyksien löytäminen käy vaikeaksi. Kurssit ja alan kirjat keskittyvät valitettavan usein näpertelyyn pienten yksityiskohtien kanssa niin, että taustat ja yhteydet jäävät hämäriksi. Alan nopea kehitys taas aiheuttaa sen, että eilen opittu tieto on tänään jo vanhentunutta.

Lisäksi tietotekniikka ja sen tuottamat ilmiöt poikkeavat monista reaalimaailman vastineistaan. Tämä vaikeuttaa asioiden hahmottamista sitä enemmän, mitä kokeneempi (lue: vanhempi) henkilö on kyseessä. Tietotekniikan esineet voivat näyttää tutuilta, mutta niissä pätevät uudet lait. Esimerkiksi hakemistopuu voi näyttää puulta, mutta sillä ei ole mitään

tekemistä puun kanssa. Ei edes arkiston kanssa, mapeista nyt puhumattakaan. Hakemistopuu on täysin kuvitteellinen — suorastaan virtuaalinen — keksintö. Sen toiminta on helpointa omaksua kun unohtaa kaiken aiemmin opitun.

Tietotekniikka on helppoa, kun sitä vertaa vaikkapa maatalous- ja pakolaispolitiikkaan tai lasten kasvatukseen — miesten ja naisten välisistä suhteista nyt puhumattakaan. Nämä asiat ovat vaikeita, koska niihin ei ole olemassa valmiita ja yksikäsitteisiä vastauksia. Asiat ovat monimutkaisia ja niihin vaikuttavat inhimilliset tekijät, joiden merkityksiä ja seurauksia on vaikeata arvioida. Yhtä oikeata vastausta ei ole. Jos asiaa kysyy kymmeneltä eri henkilöltä, saa ainakin kymmenen erilaista ja keskenään ristiriitaista vastausta.

Älä silti kerro kenellekään, miten helppoa tietotekniikka on! Jos tieto pääsisi leviämään, me alalla olevat jäisimme ilman työtä ja se sädekehä, jonka saat työtoveriesi ja tuttaviesi piirissä osaamalla tässä kirjassa kerrotut niksit, jekut ja muut asiat, katoaisi...

1

Tarkkaile työasentoasi

Tietokone- ja näyttöpäätetyön yleistyminen on tuonut esille myös työhön liittyvät vaaratekijät. Vaikka näppäimistön ja näyttölaitteen parissa tehtävä työ onkin kevyttä, siinä saattaa olla omat terveysriskinsä. Istumatyö, jossa tehdään vain pieniä mutta tarkkuutta ja keskittymistä vaativia liikkeitä, on toistuvuutensa vuoksi rasittavaa ja edellyttää säännöllisiä verryttely- ja lepotaukoja.

Terveysriskeistä eniten huomiota on saanut näyttöpäätteen aiheuttama säteily, kenties siksi, että jo sana säteily saa maallikon karvat nousemaan pystyyn. Näyttöpäätteen säteilyllä ei kuitenkaan ole mitään tekemistä esimerkiksi ydinvoimaloiden säteilyn kanssa. Näyttölaitteiden yhteydessä pitäisikin puhua sähkö- ja magneettikentistä eikä säteilystä.

Tilanne on sikäli ongelmallinen, että kukaan ei tiedä miten ihminen reagoi heikkoihin, mutta jatkuviin kenttiin. Niitähän esiintyy kaikkialla, esimerkiksi kotona televisiota katseltaessa. Työsuojelun edelläkävijänä

ovat jälleen toimineet ruotsalaiset, jotka ovat luoneet MPR-nimellä kulkevat raja-arvot sallitulle näyttöpäätesäteilylle. Alkuperäistä MPR-luokitusta kiristettiin myöhemmin MPR II-arvoissa.

Ongelma on vain siinä, että MPR-arvot ovat pelkkiä lukuja, jotka eivät perustu mihinkään fysiologisiin tuloksiin. Kukaan ei ole todistanut, että MPR-normit alittavat näytöt olisivat turvallisia tai toisinpäin, että arvot ylittävistä näytöistä olisi haittaa.

Näyttösäteilyä suurempi ja todellisempi ongelma ovatkin lihasten jatkuvasta jännittämisestä ja virheellisestä työskentelyasennosta aiheutuvat vaivat, kuten niskan ja selän rasittuminen. Myös kädet ja ranteet ovat vaarassa, mikäli kirjoitusasento tai kirjoitustekniikka on huono. Vaivoilla on nimetkin: karpaalitunnelisyndrooma ja toistuvan rasituksen vamma. Toisin kuin näyttöpäätesäteily, ne on tunnettu jo kauan, sillä esimerkiksi viulunsoittajat ovat kärsineet samoista ongelmista.

Miten siis välttyä ongelmilta? Näppäimistön pitää olla riittävän alhaalla eivätkä ranteet saa olla taivutettuina kirjoituksen aikana. Vähiten jäseniä rasittaa sokkokirjoitus, jossa katse kohdistuu näyttöön tai paperiin ja sormet tekevät omaa työtään. Näppäimistön eteen sijoitettava rannetuki kohottaa ranteita sopivasti niin, että ne eivät pääse taipumaan.

Normaalin ergonomian lisäksi kannattaa huolehtia säännöllisestä taukojen pitämisestä ja jäsenten verryttelystä. Silmiä voi lepuuttaa katsomalla ikkunasta ulos ja kohdistamalla katse horisonttiin. Sormia voi rentouttaa voimistelemalla niitä ja ravistelemalla käsiä.

Ja varmuuden vuoksi kannattaa valita omaan mikroonsa matalasäteilynäyttö, jos mahdollista.

2

Älä turhaan sammuta mikroa

Koska mikrossa käytetyt osat ovat pääasiassa elektronisia komponentteja, ne eivät kulu käytössä. Jatkuvaa käyttöä vahingollisempaa on toistuva sammuttaminen ja päällekytkeminen, koska tällöin komponentit vuoroin laajenevat ja vuoroin supistuvat. Lämpötilaero saattaa olla jopa yli 50 astetta, mikä aiheuttaa mekaanista rasitusta komponenteille.

Erityisen kuluttavaa on päällekytkeminen. Ei ihme, että monet viat syntyvätkin juuri sähköjä kytkettäessä. Ilmiö on jokaiselle tuttu sähkölampuista, joilla on tapana poksahtaa rikki juuri valoja sytytettäessä. On melko harvinaista, että lamppu sammuu kesken käytön.

Miten mikron kanssa sitten pitäisi menetellä? Jatkuva päälläolo kuormittaa kiintolevyä ja tuuletinta, koska niiden pitää pyöriä koko ajan. Sähkön kytkeminen ja sammuttaminen puolestaan kuormittaa elektronisia osia. Järkevin kompromissi lieneekin sammuttaa mikro viikonlopuiksi ja vaikka yöksikin, mutta päivän kuluessa sähköjä ei kannata katkaista. Lähiverkon serverikoneita ei kannata sammuttaa koskaan. Näyttölaite kannattaa sammuttaa, sillä ruudunsäästäjästä huolimatta päälläolo kuluttaa kuvaputken fosforia.

Jos asiaa ajatellaan laajemmin huomataan, että mikrojen turhalla käynnissä pitämisellä on myös kansantaloudellista merkitystä. Oletetaan, että mikro kuluttaa päällä ollessaan 100 W tehoa. Jos 200000 mikroa on käynnissä 10 turhaa tuntia vuorokaudessa, ne kuluttavat vuositasolla 72000 MWh ylimääräistä energiaa. Jos yksi kilowattitunti maksaa 40 penniä, tämä vastaa 28,8 miljoonan markan ylimääräistä sähkölaskua.

3

Miksi kilo on 1024 eikä 1000?

Aloitteleva mikronkäyttäjä oppii piankin, että mikroista puhuttaessa kilo tarkoittaa 1024 eikä 1000. Mutta miksi näin? Miksi kilon pitää poiketa siitä, mitä ruokakaupassa käytetään?

Koska ATK on alusta pitäen ihmisten kehittämää, myös kilon suuruus on pelkkä sopimuskysymys. Se on sovittu 1024:ksi ainoastaan siksi, että käyttäjien elämä olisi helpompaa. Koska 1024 on tietokoneen kannalta tasaluku mutta 1000 ei, kilosta tehty sopimus saa monet tärkeät luvut näyttämään helpommilta ja tekee niistä lyhyempiä kirjoittaa.

Esimerkiksi se määrä perusmuistia, joka PC:hen voidaan asentaa, on tarkkaan ottaen 655360 tavua. Jos kilo olisi määritelty tuhanneksi, määrä voitaisiin ilmaista sanomalla "655,36 kilotavua" eikä kilo-

etuliitteestä olisi juuri mitään hyötyä. Mutta kun kilo määritellään 1024:ksi, monet tietokoneen kannalta tärkeät luvut menevät siihen mukavasti tasan. Esimerkiksi 655360 on sama kuin 1024 x 640 joten muistimäärä supistuu kauniisti "640 kilotavuksi". Varjopuolena on, ettei tavujen tarkka määrä käy ilmi pelkällä vilkaisulla vaan vaatii aukilaskemisen. Toisaalta tarkalla määrällä ei useinkaan ole väliä — tärkeintä on, että luvut ovat lyhyitä ja ne on helppo lausua ja kirjoittaa.

Miksi sitten PC:hen saa 655360 tavua muistia eikä esimerkiksi 600000 tavua?

Se on suoraa seurausta tietokoneen toimintaperiaatteesta, joka perustuu ykkösiin ja nolliin. Jokainen uusi bitti tai osoitelinja kaksinkertaistaa käytettävissä olevat muistimäärät. Tärkeitä lukuja ovat siten 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 ja niin edelleen.

4

Paljonko on mega?

Paljonko sitten on mega? Onko se 100000, 1024000 (=1000*1024) vai 1048576 (=1024*1024)? Asia ei olekaan ihan niin yksinkertainen kuin voisi kuvitella, sillä megan tarkka arvo riippuu asiayhteydestä.

Silloin kun on kyse keskusmuistin määrästä, mega on kilon verran kiloja eli 1048576. PC-mikro, jossa on kaksi megatavua muistia, sisältää siis 2 * 1048576 = 2097152 tavua muistia. 286- ja 386SX-mikrojen enimmäisraja on 16 megatavua eli 16777216 tavua. Ja niin edelleen. Tämä lukujen harmonia aiheutuu siitä, että prosessorin käyttämässä binäärijärjestelmässä jokainen lisäbitti (tai muistin tapauksessa osoitelinja) kaksinkertaistaa käytettävissä olevat mahdollisuudet.

Asia muuttuu mutkikkaammaksi heti kun siirrytään massamuisteihin. Jotta myyjän lupaama "kahdensadan megan kiintolevy" todella olisi nimensä veroinen, sen pitäisi pystyä tallentamaan 209715200 tavua eli merkkiä. Usein tällaiset levyt myydään jo "210-megaisena" vaikka todellinen arvo megatavuissa mitattuna onkin tasan kaksi sataa. Myyjät (tai laitteistaan ylpeät käyttäjät) unohtavat mielellään megatavun tarkan arvon, koska siten luvut saadaan näyttämään isommilta.

Levykkeillä tilanne on vielä sekavampi. Vaikka yleisesti sanotaankin, että HD-korpulle mahtuu 1,44 megatavua, se ei tarkkaan ottaen pidä paikkaansa. HD-korpun tarkka kapasiteetti saadaan kertomalla puolten määrä (2), sektorien määrä (18) ja urien määrä (80) keskenään ja ottamalla huomioon, että jokaiseen lohkoon mahtuu 512 tavua eli puoli kiloa. Kapasiteetiksi saadaan siten 2*18*80/2 eli 1440 kilotavua. Tämä on kuitenkin aivan eri asia kuin 1,44 megatavua, sillä 1440*1024 = 1474560 kun taas 1,44*1024*1024 = 1509949,44! Jos siis HD-korpun kapasiteetti halutaan ilmaista megatavuissa, pitäisi käyttää lukua 1,40625 megatavua. Ei ihme, että tarkkaan laskien virheellinen lukuarvo 1,44 elää sitkeästi käytössä.

Mega-etuliitteen fysikaalinen merkitys on tasan miljoona. Tällaisena sitä käytetään vieläkin esimerkiksi kellotaajuutta ilmaistaessa. Mikro, jonka kellotaajuus on 33 megahertsiä, suorittaa 33000000 kellojaksoa sekunnissa.

Ja kuten kaikki tietävät, megan merkki on iso M-kirjain. Pieni m tarkoittaa milliä eli tuhannesosaa. Jos kiintolevyn kapasiteetiksi ilmoitetaan 40 mb, se tarkoittaa 40 millibittiä eli 0,04 bittiä. Se taas on 0,005 tavua. Eli todella vähän!

5

Iso vai pieni K?

Sekaannusta tuntuu vallitsevan myös siitä, onko kilo-etuliitteen merkki iso K vai pieni k. Varsinkin mainoksissa ja esitteissä näkee usein merkintöjä tyyliin "muistia 640 KB", mikä on selvästi väärin, sillä K tarkoittaa Kelviniä, mikä puolestaan on lämpötilan yksikkö.

Silloin kun puhutaan etuliitteistä, kilon merkki on aina pieni k.

6

B:llä ja b:llä on eroa

Ison ja pienen K:n välillä on eroa, mutta kyse on lähinnä kauneusvirheestä, sillä kukapa nyt sekoittaisi lämpötilan ja muistin määrän keskenään.

Bitin ja tavun välillä on kuitenkin iso ero ja valitettavasti niitä molempia merkitään samalla kirjaimella. Väärinkäsitysten välttämiseksi on voimassa sääntö, että iso B tarkoittaa tavua ja pieni b bittiä. Siten merkintä kB tarkoittaa kilotavua ja kb kilobittiä. Kaksinkertaisesti virheellinen merkintä Kb on valitettavan yleinen. Ehkä sitä käytetään siksi, että se niin läheisesti muistuttaa Oy Ab-lyhennettä.

Koska yhteen tavuun mahtuu kahdeksan bittiä, muistimäärästä kertovat luvut muuttuvat kahdeksankertaisiksi kun ne ilmoitetaan tavujen sijasta bitteinä. Esimerkiksi 640 kilotavua (kB) onkin sama kuin 5120 kilobittiä (kb) eli auki laskettuna 5242880 bittiä.

Suoranainen sekaannuksen vaara syntyy silloin, kun ilmoitetaan siirtonopeuksia. Esimerkiksi levyohjaimen tai modeemin siirtonopeus voidaan ilmoittaa joko bitteinä tai tavuina sekunnissa. Mainosmiehet käyttävät mielellään bittejä, koska luvut näyttävät silloin selvästi suuremmilta. Esimerkiksi kiintolevyohjain, joka pystyy siirtämään 800 kilotavua (800 kB/s) sekunnissa voidaan ilmoittaa myös 3,6 megabittiä sekunnissa (3,6 Mb/s) jotta nopeus näyttäisi suuremmalta. Jos ison ja pienen b-kirjaimen välillä ei tehdä tarkkaa eroa, mainoksen lukija ei voi tietää kumpaa tarkoitetaan.

Jos b-kirjainten erottelu tuntuu vaikealta, voi väärinkäsitysten välttämiseksi käyttää suomalaista t-lyhennettä tavun sijaan. Bitin puolestaan voi kirjoittaa auki, jolloin 100 kb/s kirjoitetaan muodossa 100 kbit/s eikä sekaannuksen vaaraa enää ole.

7

Baudi ei ole sama kuin bps

Modeemien siirtonopeus ilmoitetaan yleensä bitteinä tai kilobitteinä sekunnissa. Tavallisen perusmodeemin nopeus on 2400 bittiä sekunnissa (bps), mutta uudemmat mallit pystyvät siirtämään jopa 9600 tai 14400 bittiä sekunnissa.

Tämä arvo on aivan eri asia kuin baudi, jota varsinkin amerikkalaiset käyttävät säännöllisesti väärin. Baudi on signaalimuutosten yksikkö ja yleisessä puhelinverkossa käytetty tekniikka asettaa teknisen ylärajan baudien määrälle. Tämä raja on hyvin lähellä 2400:tta.

Vanhat modeemit koodasivat jokaiseen signaalimuutokseen vain yhden bitin, jolloin bittinopeus oli sama kuin baudinopeus. Uudemmat mallit siirtävät jokaisella signaalilla kuitenkin useita bittejä, jolloin bittinopeus on moninkertainen baudinopeuteen verrattuna. Esimerkiksi V.32-tyyppinen modeemi siirtää 2400 baudin nopeudella 9600 bittiä sekunnissa, koska jokaisella signaalilla siirretään neljä bittiä.

Varsinkin modeemiharrastajat ovat tarkkoja baudin ja bps:n eroista. Jos erehtyy käyttämään baudi-termiä muiden kuullen saa osakseen paitsi sääliviä katseita myös ikuisen aloittelijan maineen.

Seuraavassa taulukossa on lueteltu eri modeemistandardeissa käytetyt baudi- ja bittinopeudet:

	baudia	bps
V.21	300	300
V.22	600	1 200
V.22bis	600	2 400
V.32	2 400	9 600
V.32bis	2 400	14 400

8

Jatkomuistin tarkka määrä

Kun käyttäjä ostaa mikron, siinä saattaa mainoksen mukaan olla "neljä megatavua RAMia". Mutta missä tuo muisti on?

Muistin kokonaismäärä on siis neljä megatavua eli 4096 kilotavua. Siitä perusmuistia on 640 kiloa joten jäljelle jää vielä 3456 kiloa. Yleensä tämä määrä on jatkomuistia (extended), jolloin sitä voidaan käyttää levyn välimuistina (esimerkiksi Smartdrv:n kanssa) tai RAM-levynä (Ramdriven kanssa). DOSia kehittyneemmät käyttöjärjestelmät, kuten Windows ja OS/2, pystyvät käyttämään muistia suoraan.

Tästä huolimatta muistin määrän näyttävät ohjelmat (kuten koneen oma SETUP tai DOSin MEM) saattavat näyttää jatkomuistin kooksi vähemmän kuin laskemalla saatu 3456 kiloa. Mihin siis loppu muistista on hävinnyt? Onko se ylämuistissa? Ei.

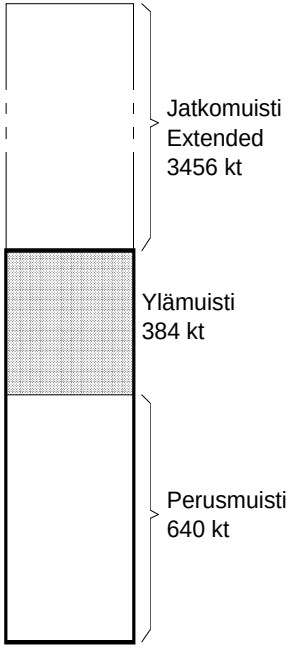
Ylämuistia, joka jää perusmuistin ja jatkomuistin väliin, *ei koskaan* lasketa muistimääriin mukaan, koska alue ei nimestään huolimatta ole varsinaista muistia. Suuri osa siitä on pelkkää tyhjää: muistiosoitteet ovat olemassa, mutta niissä ei ole muistipiirejä. Jos kone on vähintään 386-tasoa, osa jatkomuistista voidaan siirtää ylämuistialueen tyhjiin kohtaan ja näin saada tämäkin alue hyödynnettyä, mutta se on kokonaan oma tarinansa eikä sitä ole koskaan tehty valmiiksi. Se vaatii aina CONFIG.SYSin muokkausta ja apuohjelmien käyttöä.

Tyhjän lisäksi ylämuistialueella on näyttökortilta tuleva grafiikkamuisti, joka näkyy omassa 64 kilotavun ikkunassaan sekä koneen oma BIOS ROM, joka on sekin yleensä 64 kilotavua. IBM:n koneissa BIOS ROM saattaa olla tuplasti isompi, jolloin ylämuistiin jää vähemmän tyhjää tilaa.

Muut lisäkortit, kuten CD ROM- ja SCSI-ohjaimet tai lähiverkkokortit saattavat sisältää omia ROM-alueita, jotka myös täyttävät ylämuistialueessa muuten olevaa tyhjyyttä.

Missä puuttuva muisti sitten on?

Koneesta riippuen osa siitä (64-256 kilotavua) voi olla ns. varjomuistia (Shadow RAM), jolloin mikro siirtää jatkomuistista alueita



Jos mikrossa on neljä megatavua (4096 kt) muistia, siinä on 640 kilotavua perusmuistia ja 3456 kiloa jatkomuistia. Eräissä koneissa osa jatkomuistista on varattu varjomuistiksi, jolloin vapaan jatkomuistin määrä näyttää pienemmältä. Ylämuistialuetta ei silti koskaan lasketa muistialueisiin.

ylämuistin BIOS ROMin ja näyttökortin ROMin päälle ja kopioi niiden sisällön itseensä. Koska jatkomuisti on nopeata 32-bittistä muistia, siihen kopioidut ROM-rutiinit toimivat paljon nopeammin kuin alkuperäisiltä ROM-piireiltä ajettuina. Kopioinnin ansiosta ne näyttävät sijaitsevan entisillä paikoillaan.

Jos laitteen suunnittelija on halunnut päästä vähällä, muisti voi yksinkertaisesti puuttua. Tällöin perusmuistin 640 kilotavun lisäksi näkyy vain 3072 kilotavua jatkomuistia. Puuttuva 384 kilotavua on jäänyt ylämuistin "alle", eikä sitä voida mitenkään käyttää. Onneksi tällaiset laiteratkaisut ovat käyneet harvinaisiksi, vaikka ne vielä 286-koneissa olivatkin yleisesti käytössä.

9

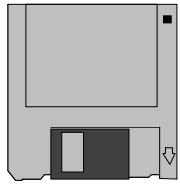
Mistä näkee korpun kapasiteetin?

Korppuja valmistetaan kolmelle eri tallennuskapasiteetille:

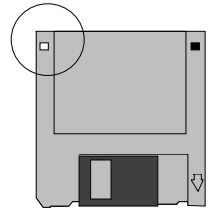
DD	720 kt
HD	1,44 Mt
ED	2,88 Mt

Taulukossa ilmoitettu kapasiteetti on se määrä tavuja, joka levykkeelle voidaan tallentaa levykkeen alustamisen jälkeen. Levykkeessä saattaa lukea suurempikin määrä kuten HD-korpuissa "2.0 MB", mutta käytön vaatimat sektori-, ohjaus- ja tarkistussummamerkit vievät oman tilansa.

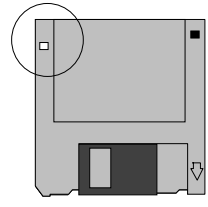
Jotta levykeasema voisi erottaa, millaisesta levykkeestä on kyse, korppuihin on lisätty tunnistusaukko. Jos aukko puuttuu, korppu on



DD (720 k)



HD (1,44 M)



ED (2,88 M)

Korpun tyyppi on helppo tunnistaa aukkojen määrästä ja niiden sijainnista. ED-korpuissa tunnisteaukko on alempana kuin HD-korpussa.

DD-tyyppiä. Kahden aukon korppu on joko HD- tai ED-tyyppiä, mutta ED-korpussa aukko sijaitsee hieman alempana kuin HD-korpussa.

Koska HD-korput ovat jonkin verran DD-korppuja kalliimpia, jotkut ovat ryhtyneet rei'ittämään DD-korppuja niin, että ne näyttävät mikrolle HD-korpuilta. Tämä on kuitenkin väärää säästämistä, joka saattaa kostautua tärkeiden tiedostojen menetyksenä. DD- ja HD-korppujen magneettinen materiaali on erilaista, joten keinotekoisesti 1,44-megaiseksi muutettu DD-korppu ei toimi luotettavasti. Lisäksi rei'itys saattaa jättää levyn pinnalle muovin murusia, jotka vahingoittavat levyn pintaa sen pyöriessä.

10

IBM:n mikrot alustavat DD-korput väärin

Jostain syystä IBM on käyttänyt mikromalleissaan korppuasemia, jotka eivät tutki lainkaan levykkeen tyyppiä. Jos asema on 1,44-megainen, asemaan laitetaan 720-kilon DD-korppu ja kirjoitetaan

```
C:\>FORMAT A:
```

mikro alustaa levykkeen kaikessa hiljaisuudessa 1,44-megaiseksi. Mikään oikein toimiva levykeasema ei kuitenkaan pysty lukemaan sitä, koska ne tunnistavat korpun DD-tyyppiseksi ja yrittävät lukea sitä 720 kilotavun mukaan. Samanlaista asemaa saattaa esiintyä myös joidenkin muiden laitevalmistajien yksittäisissä malleissa, mutta vain IBM:llä aseman käyttö on ollut järjestelmällistä.

Jos asema toimii oikein, FORMAT-ohjelma antaa virheilmoituksen ja huomauttaa, ettei se pysty alustamaan levykettä 1,44-megaiseksi:

```
Invalid media or Track 0 bad - disk unusable.
Format terminated.
```

Ilmoituksessa esiintyvällä nollauralla ei ole mitään tekemistä tämän asian kanssa.

Tällä komennolla on myös helppo testata, onko oman korppuaseman tunnistus kunnossa.

Jos tunnistus ei toimi ja halutaan, että levykkeitä pystyy lukemaan myös muilla koneilla, DD-levykkeitä alustettaessa pitää käyttää komentoa

```
C:\>FORMAT A: /F:720
```

Tämä varmistaa, että levykkeet tulee alustettua niiden nimelliskapasiteetin mukaisesti 720-kiloisiksi, jotta muidenkin mikrojen käyttäjät pystyvät lukemaan niitä. Varsinkin PS/2-käyttäjien kannattaa painaa komento mieleensä!

11

BIOSin päiväys ja laitetyyppi

Useimmissa mikroissa BIOS ROM sisältää päivämäärämerkinnän, josta voi päätellä jotain koneen iästä. Ellei BIOSia ole vaihdettu jälkeenpäin uudempaan, kone on todennäköisesti samaa ikäluokkaa kuin BIOSin päiväys osoittaa.

Eräät ohjelmat tai lisäkortit saattavat joissakin koneissa vaatia vähintään tietyn päiväyksen tai sitä uudemman toimiakseen yhdessä. Siksi on hyvä tietää, mistä päiväyksen voi tarkistaa.

Jos BIOSissa on päiväysmerkintä, se saadaan näkyviin käynnistämällä DEBUG ja antamalla komento

```
-d FFFF:5 L 8
FFFF:0000   30 34 2F-31 36 2F 38 38         04/16/88
```

Päiväyskoodi on tallennettu amerikkalaiseen tapaan kuukausi/päivä/vuosi.

Komento

```
-d F000:FFFE L 1
F000:FFF0                                           FC
```

näyttää päiväyksen jäljessä tulevan tyyppikoodin, jota IBM noudatti ensimmäisissä laitteissaan. Kun muut valmistajat alkoivat tehdä omia, IBM:n kanssa yhteensopivia mikroja, ne alkoivat käyttää FC-koodia yhteensopivuuden varmistamiseksi ja sen jälkeen koko tavu menetti merkityksensä. Nykyisin kaikki vähintään 286-tasoa olevat mikrot käyttävät koodia FC.

IBM:n alkuperäiset tyyppikoodit olivat:

FF	perus-PC
FE	XT
FD	PC Junior
FC	AT
FB	PC XT 286
FA	PS/2 30
F9	IBM Convertible
F8	PS/2 80

Eräät diagnostiikkaohjelmat saattavat näyttää myös toisen alikoodin, jota ne kutsuvat nimellä subtype. Esimerkiksi IBM L40 matkamikrossa alikoodi on 23 ja VP-mallisarjassa 4A. Useimmissa muissa mikroissa se on 1. Alikoodi ei kuitenkaan sijaitse missään vakiopaikassa, joten sitä ei voi tarkistaa mitenkään helposti. Diagnostiikkaohjelmat selvittävät koodin BIOSin taulukosta sijoittamalla AH-rekisteriin arvon C0 ja suorittamalla INT 15:n. Sen jälkeen ES:BX-rekisterissä palautuvan osoitteen neljäs tavu on alakoodi.

Myös monien grafiikkakorttien BIOS-piireissä on päiväysmerkintä. Usein niissä lukee myös kortin nimi ja valmistaja, jota tietoa diagnostiikkaohjelmat käyttävät hyväkseen. Jos päiväys on olemassa ja grafiikkakortin ROM sijaitsee tavallisella paikallaan C0000:sta alkaen, päiväys saadaan näkyviin komennolla

```
-d C000:9 L 8
C000:0000   30 32 2F 32 36 2F 38 38        02/26/88
```

Grafiikka-ROMin alussa voi näkyä muutakin mielenkiintoista. Sana "IBM" mainitaan muodon vuoksi, koska PC-kauden alussa oli ohjelmia, jotka suostuivat toimimaan vain aidoissa IBM PC:issä ja ne tunnistivat

laitteen juuri tällä paikalla esiintyvästä IBM-nimestä. Tärkeintä on, että kirjaimet I, B ja M löytyvät ROMista juuri tietystä osoitteesta alkaen.

Paradisen vanha VGA-kortti sisältää jopa tarkoituksella väärin kirjoitetun yhteensopivuutta tarkoittavan sanan heti "IBM" merkin jälkeen:

```
-d C000:0
C000:0000  55 AA 30 EB 15 37 34 30-30 30 32 2F 32
36 2F 38   U.0..740002/26/8
C000:0010  38 2D 31 30 3A 34 35 3A-34 30 E9 A8 00
00 49 42   8-10:45:40....IB
C000:0020  4D 20 43 4F 4D 50 41 54-41 42 4C 45 00
00 00 00   M COMPATABLE....
C000:0030  00 00 00 00 00 30 30 33-30 35 36 2D 30
30 32 43   .....003056-002C
C000:0040  4F 50 59 52 49 47 48 54-20 50 41 52 41
44 49 53   OPYRIGHT PARADIS
C000:0050  45 20 53 59 53 54 45 4D-53 20 49 4E 43
2E 20 31   E SYSTEMS INC. 1
```

BIOS ROMin selaaminen (osoitteesta F000:0000 alkaen) saattaa myös paljastaa mielenkiintoisia asioita, joten siitä vain tutkimaan!

Asentaminen ja toiminta-asetukset

12

Varmista FILES-asetuksen riittävyys

Eräiden DOS-versioiden oma asennusohjelma kirjoittaa CONFIG.SYSiin liian pienen FILES-asetuksen. Kun sovellukset eivät pysty avaamaan kaikkia tarvitsemiaan tiedostoja, ne antavat epämääräisiä virheilmoituksia tai saattavat toimia väärin ilman näkyvää syytä. Kaikki sovellukset eivät ole yhtä täsmällisiä kuin WordPerfect 5.1, joka ilmoittaa napakasti että "Tiedostokahvojen määrä ei riitä" mikäli FILES-asetus on liian pieni.

WordPerfect 5.1 vaatii vähintään 15 tiedostokahvaa käynnistyäkseen ilman virheilmoitusta. Koska kahvat eivät vie juuri lainkaan muistia, arvo kannattaa varmuuden vuoksi kirjoittaa yläkanttiin. Pelkkiä DOS-ohjelmia ajettaessa riittävä arvo on 20, mutta Windows-käytössä sitä voi kasvattaa aina 40 tai 50 asti. Esimerkiksi Microsoftin Access-tietokantaohjelma suosittelee vähintään 50 tiedostokahvaa.

Eräät Windowsin käytössä esiintyvät ongelmat saattavat myös ratketa kasvattamalla puskuriarvoa esimerkiksi sataan.

13

Poista tarpeettomat puskurit

FILES-asetus on helppo optimoida. Tiedostopuskurien määrän kertova BUFFERS on paljon hankalampi tapaus, koska sopiva asetus riippuu sekä DOSin että välimuistin versioista.

Aloitetaan vanhimmasta päästä. Jos DOS on versiota 3.3 tai sitä vanhempi eikä välimuistiohjelmaa ole käytössä, sopiva puskuriarvo on 15-20. DOS 4.0:sta alkaen puskureita voi laittaa lisääkin ilman, että levyn nopeus siitä hidastuu, mutta jokainen puskuri kuluttaa puoli kilotavua kallisarvoista perusmuistia.

DOS 4.0:sta alkaen voi kokeilla myös ennakkopuskurien vaikutusta. Esimerkiksi rivi

```
BUFFERS=20,4
```

ottaa käyttöön 20 tavallista puskuria sekä neljä ennakkopuskuria, joihin luetaan jokaisella lukukäskyllä neljä seuraavaa levyn lohkoa. Yleensä ennakkopuskurien vaikutus on kuitenkin marginaalinen ja saattaa jopa hidastaa levykäsittelyä, mikäli levyä luetaan satunnaisesti sieltä täältä.

Välimuisti muuttaa kaiken tämän. Se tekee saman työn kuin puskuritkin, mutta tehokkaammin. Puskurien määrää voi siis vähentää, mutta lopullinen arvo riippuu Smartdrv:n versiosta. Versiosta 4.0 alkaen Smartdrv osaa nopeuttaa myös levykkeiden käsittelyä, joten puskuriasetuksen voi kirjoittaa vaikkapa muotoon

```
BUFFERS=4
```

Vanhan Smartdrv:n (versio < 4.0) tuntee siitä, että se asennetaan CONFIG.SYSiin muiden laiteohjainten tapaan. Tämä Smartdrv ei osaa käsitellä levykkeitä, joten puskuriasetuksen voimakas pienentäminen hidastaa levykkeiden kopiointia. Silloin asetus kannattaa kirjoittaa esimerkiksi seuraavasti:

```
BUFFERS=8
```

Ennakkopuskureita ei kannata varata, koska Smartdrv hoitaa niiden tehtävän tehokkaammin.

Koska puskurit varataan DOS 5.0:sta alkaen HMA-alueelta eikä perus- tai ylämuistista, suurempikaan puskurimäärä ei vähennä sovelluksille jäävää muistia. Lukuisista puskureista ei kuitenkaan ole vastaavaa hyötyä, sillä välimuisti hoitaa saman asian tehokkaammin.

14

FASTOPEN on tarpeeton

DOS 3.3:sta alkaen DOSiin lisättiin FASTOPEN-nimellä kulkeva apuohjelma. Sen tarkoitus on nopeuttaa tiedostojenkäsittelyä niin, että ohjelma pitää taulukkoa avatuista tiedostoista. Jos samaan hakemistoon tai samoihin tiedostoihin viitataan myöhemmin uudelleen, niiden sijainti levyllä löytyy taulukosta eikä DOSin tarvitse kahlata hakemistopuuta läpi. Tämä nopeuttaa toimintaa silloin, kun ohjelmat avaavat useita tiedostoja, kun hakemistorakenne on erityisen monimutkainen tai kun samoja tiedostoja avataan toistuvasti.

Käytännössä FASTOPENista saatava hyöty on marginaalinen, varsinkin silloin kun käytetään välimuistia. Koska FASTOPEN kuluttaa helposti yli 10 kilotavua kallisarvoista keskusmuistia, se kannattaa jättää pois AUTOEXEC.BATista, vaikka ainakin eräät DOS 4.0:n asennusohjelmat pyrkivätkin lisäämään sen sinne automaattisesti.

15

SETVERiä tarvitaan tuskin koskaan

DOSin asennusohjelma lisää 5.0-versiosta alkaen CONFIG.SYSiin pienen laiteohjaimen, jonka tarkennin on poikkeuksellisesti EXE:

```
DEVICE=C:\DOS\SETVER.EXE
```

Laiteohjaimen tehtävänä on valehdella eräille sovelluksille, että käytössä on todellista vanhempi DOS-versio. Koska tällaiset sovellukset ovat nykyään erittäin harvinaisia, SETVERiäkään ei tarvita juuri koskaan. Toisaalta sen käytöstä ei ole suurta haittaakaan, koska muistiin ladattuna SETVER vie vain noin 400 tavua muistia.

SETVERin tarve on helppo selvittää. Pelkkä komento

```
C:\>SETVER
```

näyttää, mitkä ohjelmat saavat SETVERin ansiosta todellista vanhemman DOS-version. Jos koneessa ei käytetä yhtään listassa lueteltua ohjelmaa ei SETVERiäkään tarvita.

15 B

Valitettavasti SETVERin avulla voidaan tehdä myös vahinkoja — joko tarkoituksella tai epähuomiossa. Esimerkiksi komento

```
SETVER COMMAND.COM 3.0
```

saa käyttöjärjestelmän näkemään komentotulkin versiona 3.0. Jos käytössä on jokin muu DOS-versio (kuten ilmeisesti on, koska SETVERiä voidaan käyttää vasta DOS 5.0:sta alkaen!), tämä estää mikroa käynnistymästä. Alkulatauksen aikana ruudulle tulostuu varoitus väärästä DOS-versiosta ja mikron saa käyntiin vasta DOS-levykkeen avulla.

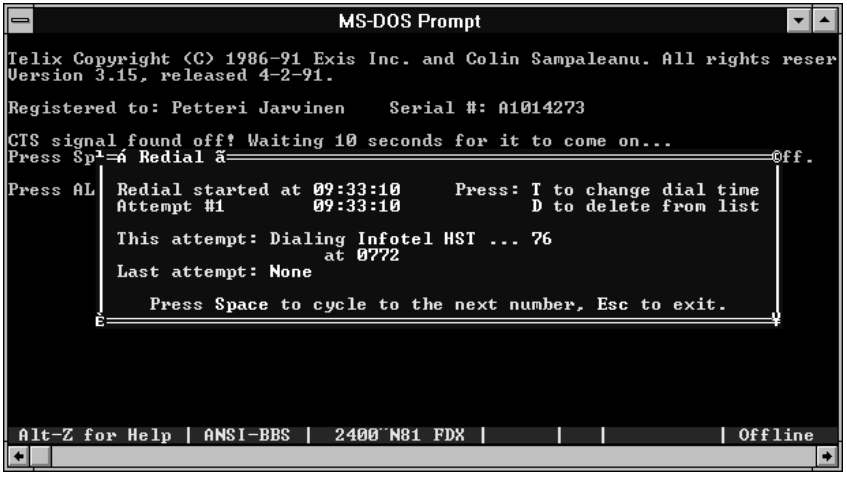
Koska SETVER on tarpeen vain harvoissa tapauksissa, sen rivi kannattaa poistaa CONFIG.SYSistä. Vaikka ohjelma olisikin levyllä, SETVER-komennoista ei ole vaaraa koska ne aktivoituvat vain CONFIG.SYSissä ladattavana laiteohjaimena.

16

Tarpeettomat koodisivut

Koodisivut otettiin käyttöön DOS 3.2:ssa. Niiden tarkoituksena on mahdollistaa tavallista useampien kansallisten erikoismerkkien käyttö. Koska merkkien määrä on kuitenkin rajoitettu, uudet merkit on jouduttu sijoittamaan eräiden puoligraafisten merkkien päälle.

Me suomalaiset emme tarvitse koodisivuja oikeastaan koskaan. Ne kuluttavat turhaan muistia ja saavat puoligraafisia merkkejä käyttävät sovellukset näkymään väärin. DOSin asennusohjelma ottaa koodisivut kuitenkin käyttöön, koska se kuvittelee kaikkien USAn ulkopuolisten maiden tarvitsevan niitä. Näin ei kuitenkaan ole.



Jos koodisivuja käytetään, eräät puoligraafiset erikoismerkit korvautuvat tekstimerkeillä mikä saattaa sotkea joidenkin ohjelmien näyttöä. Lisäksi koodisivujen käyttö kuluttaa kallisarvoista muistia. Sama ilmiö näkyy niin aidossa DOSissa kuin Windowsin DOS-tilassakin. Koodisivujen poistaminen Windows-ikkunasta tapahtuu kuitenkin eri tavalla kuin DOSista.

Kun DOS asennetaan sen omalla asennusohjelmalla, CONFIG.SYSiin lisätään tuki näyttölaitteen koodisivuille

```
DEVICE=C:\DOS\DISPLAY.SYS CON=(EGA,,1)
```

ja jos järjestelmään on asennettu jokin IBM:n kirjoitin, myös sille:

```
DEVICE=C:\DOS\PRINTER.SYS LPT1=(4201,,1)
```

AUTOEXEC.BATiin kirjoitetaan koodisivujen lataamiseen ja kansallisuustukeen liittyvät rivit:

```
MODE CON CODEPAGE PREPARE=((850) C:\DOS\EGA.CPI)
MODE CON CODEPAGE SELECT=850
NSLFUNC
```

Riveistä ensimmäinen lataa sivun 850 mukaiset merkit tiedostosta näyttökortin muistiin. Tiedostonimessä esiintyvä EGA tarkoittaa sekä EGAa että VGAta. Toinen rivi valitsee ladatut merkit käyttöön.

Testikoneessa ohjelmat kuluttivat DOS 5.0:lla muistia seuraavasti:

DISPLAY.SYS	8,1 k
PRINTER.SYS	14,6 k
NLSFUNC	2,7 k
---	---
Yhteensä	25,4 k

Jos koodisivuja ei haluta käyttää, kaikki nämä rivit voidaan poistaa. Paitsi että aloitustiedostot näyttävät sen jälkeen paljon selkeämmiltä, myös kallisarvoista perusmuistia vapautuu parempaan käyttöön.

Monet PC-käyttäjät jättävät koodisivurivit paikoilleen, koska eivät tiedä mitä ne tarkoittavat eivätkä siksi uskalla poistaa niitä. Ja jos rivejä alkaa poistaa yksi kerrallaan, ne antavat erilaisia virheilmoituksia yrittäessään viitata toinen toisiinsa. Mutta kun poistaa kerralla kaikki rivit, koodisivut ovat lopullisesti mennyttä.

17

Koodisivun oletusarvo vaihtui DOS 4.0:ssa

Koodisivun perusarvo määritellään CONFIG.SYSin COUNTRY-asetuksella. Kun kirjoitetaan

```
COUNTRY=358,437,C:\DOS\COUNTRY.SYS
```

koodisivuksi tulee 437, joka on mikron ROMissa valmiina oleva perussivu. Se tarjoaa kaikki ne merkit, joita suomalaiset ja ruotsalaiset mikronkäyttäjät tarvitsevat.

Koska 437 on perussivu, se on oletusarvo ja voidaan jättää merkitsemättä. Asetuksen voi siten kirjoittaa muodossa

```
COUNTRY=358,,C:\DOS\COUNTRY.SYS
```

DOS 4.0:sta alkaen oletusarvo riippuu kuitenkin maakoodista ja saa Suomen tapauksessa arvon 850.

Jos asiaa kysytään DOSilta, se väittää että käytössä on sivu 850:

```
C:\>CHCP
Active code page: 850
```

Saman vastauksen saavat myös ne sovellukset, jotka kysyvät asiaa DOSilta. Koodisivua 850 ei kuitenkaan ole ladattu käyttöön, koska lataukseen liittyvät rivit on edellisen niksin mukaisesti poistettu. Niinpä kuvaruudulle ja kirjoittimelle tulostuvat merkit *näyttävät* edelleen sivun 437 mukaisilta, kuten pitääkin.

Eräs koodisivua kysyvä ohjelma on Windowsin asennusohjelma SETUP. Kun se kysyy koodisivua DOSilta ja saa vastaukseksi 850, se kopioi tämän sivun mukaiset merkit Windowsiin. Kun DOS-ohjelmia sitten ajetaan Windowsissa ikkunoituna, niiden puoligraafiset erikoismerkit näkyvät sivun 850 mukaisesti — eli väärin.

Näiden sekaannusten välttämiseksi kannattaa DOS 4.0:sta alkaen koodisivun numero kirjoittaa aina näkyviin:

```
COUNTRY=358,437,C:\DOS\COUNTRY.SYS
```

18

SHELL-rivin /F-valitsin

SHELL-määritys, joka CONFIG.SYSissä kertoo käytettävän komentotulkin nimen (yleensä COMMAND.COM) ja sijainnin, tuntee dokumentoimattoman /F-valitsimen. Se otetaan käyttöön kirjoittamalla CONFIG.SYSiin rivi

```
SHELL=C:\DOS\COMMAND.COM /E:2000 /P /F
```

/F-valitsimen merkitys on siinä, että se vastaa automaattisesti jokaiseen Abort, Retry, Fail-kysymykseen FAIL. Tästä valitsimesta on tuskin hyötyä koti-PC:ssä, mutta silloin kun mikro toimii itsenäisesti ja ilman valvontaa, /F-valitsin estää siinä pyörivää ohjelmaa pysähtymästä esimerkiksi levyltä tulevaan virheilmoitukseen.

19

Komentotulkin sijoittaminen

Komentotulkki kannattaa yleensä sijoittaa C: aseman päähakemistoon. Vanhoissa DOS-versioissa levykkeen alustus /S-valitsimella onnistui vain, mikäli tulkki löytyi päähakemistosta. DOS 4.0:sta alkaen tätä ongelmaa ei enää ole, joten tulkin voi sijoittaa myös DOS-alihakemistoon.

Sijoittamalla komentotulkki alihakemistoon estetään yleinen vahinko, joka syntyy kun tiedostoja kopioidaan levykkeeltä kiintolevylle mutta oletushakemistoa ei ole muistettu vaihtaa. Tiedostot kopioituvat päähakemistoon ja jos niiden joukossa on levykkeellä ollut COMMAND.COM, mikro ei enää seuraavalla kerralla käynnisty ilman DOS-levykettä koska komentotulkki ei ole samaa paria käyttöjärjestelmän muiden tiedostojen kanssa.

Olipa komentotulkki missä hakemistossa tahansa, sen sijainti kannattaa kertoa heti AUTOEXEC.BATin alussa. Rivi

```
SET COMSPEC=C:\COMMAND.COM
```

kertoo komentotulkille, mistä sen pitää hakea itseään jos jokin ohjelma kirjoittaa muistissa olevan tulkin päälle. Jos ympäristömuuttujaa ei ole määritelty tai jos se osoittaa väärään paikkaan, mikron käyttö pysähtyy virheilmoitukseen

```
Cannot load Command.com, System halted.
```

COMSPEC-asetus kannattaa tehdä heti AUTOEXEC.BATin alussa, koska samassa komentojonossa saattaa olla sitä vaativia ohjelmia. Jos asetus puuttuu, mikro jumiutuu jo AUTOEXEC.BATin aikana ja ainoa tapa korjata tilanne on käynnistää kone tavallisella DOS-levykkeellä.

19 B

Komentotulkin sijoittamisella päähakemistoon on yksi selvä etu: se löytyy silloinkin, jos CONFIG.SYS jostain syystä tuhoutuu. Kun mikro käynnistyy eikä löydä CONFIG.SYSiä, se osaa etsiä komentotulkkia vain päähakemistosta. Jos tulkki on siirretty alihakemistoon, lataus pysähtyy virheilmoitukseen

```
Bad or missing Command Interpreter
```

ja ainoa tapa saada mikro käyntiin on turvautua DOS-levykkeeseen.

19 C

Kun komentotulkin sijainti kerrotaan SHELL-asetuksessa, yksi hakemistonimi riittää. Merkintä

```
SHELL=C:\DOS\COMMAND.COM C:\DOS /P /E:1000
```

on perua DOS 2:n ajoilta eikä toista hakemistoviittausta tarvita, vaan asetus voidaan kirjoittaa lyhyemmin

```
SHELL=C:\DOS\COMMAND.COM /P /E:1000
```

20

Komentotulkin korvaaminen omalla ohjelmalla

CONFIG.SYS-tiedostossa oleva SHELL-rivi kertoo, minkä nimistä komentotulkkia DOS käyttää ja mistä hakemistosta se löytyy. Yleensä tiedosto on COMMAND.COM eli DOSin mukana tuleva vakiotulkki.

Mikään ei kuitenkaan estä korvaamasta komentotulkkia omalla ohjelmalla. Kun sen nimi kirjoitetaan SHELL-riville, ohjelma käynnistyy välittömästi kun käyttöjärjestelmä on saatu ladattua. Jos sovellus lopetetaan, kone jää jumiin, sillä muistissa ei nyt olekaan komentotulkkia joka ottaisi koneen hallintaansa sovelluksen jälkeen.

Koska AUTOEXEC.BATia ei käytetä, käyttäjä ei pysty keskeyttämään ohjelman latausta. Hän ei myöskään pääse antamaan DOS-komentoja edes ohjelman lopettamisen jälkeen. Tästä voi olla suurta hyötyä silloin kun mikroa käytetään demo-ohjelman tai erityistarkoitukseen tehdyn sovelluksen ajamiseen ja halutaan, etteivät ohjelman käyttäjät pääse sotkemaan mikron levyä eivätkä kopioimaan ohjelmaa itselleen.

SHELL ei tietenkään estä koneen käynnistämistä DOS-levykkeellä. Jotta tämäkin tulisi estettyä, levykeasema pitää poistaa käytöstä joko fyysisesti tai määrittelmällä mikron SETUP niin, että käynnistystä yritetään ensin kiintolevyltä ja vasta sen jälkeen levykkeeltä.

Entä sitten ääkköset? Koska ne asetetaan paikoilleen vasta AUTOEXEC.BATissa, jota tässä niksissä ei käytetä lainkaan, ääkköset saadaan käyttöön vain siten, että CONFIG.SYSiin lisätään rivi

```
INSTALL=C:\DOS\KEYB SU,,C:\DOS\KEYBOARD.SYS
```

ennen SHELL-asetusta. Tämä edellyttää, että DOS on vähintään versiota 4.0.

21

CONFIG.SYSin vaihtaminen eri sovelluksille

Aina silloin tällöin syntyy tilanne, että jokin sovellus vaatii muista poikkeavat CONFIG.SYS-asetukset. Ainoaksi mahdollisuudeksi jää tällöin vaihtaa CONFIG.SYSiä ja käynnistää kone uudelleen. Työ kuulostaa monimutkaiselta, mutta sopivan komentojonon avulla se on mahdollista automatisoida lähes yhtä helpoksi kuin minkä tahansa ohjelman käynnistys komentojonon avulla.

Oletetaan, että käyttäjän viisi yleistä sovellusta toimivat hyvin tavallisella CONFIG.SYS-asetuksella. Yksi sovellus on kuitenkin niin vanha, ettei se tunne XMS-muistinhallintaa ja tämän vuoksi HIMEM-ohjain on poistettava käytöstä kun ohjelmaa ajetaan. CONFIG.EI sisältää CONFIG.SYS-tiedoston, joka on muuten sama kuin käytössä oleva

CONFIG.SYS, mutta siitä puuttuu DEVICE=HIMEM.SYS -rivi. CONFIG.EI tallennetaan samaan hakemistoon sovelluksen kanssa.

Sen jälkeen rakennetaan komentojono START.BAT, joka kopioi käytössä olevan CONFIG.SYSin aluksi turvaan vaikkapa nimelle CONFIG.VAR:

```
COPY \CONFIG.SYS \CONFIG.VAR
```

Näin varmistetaan, että alkuperäinen CONFIG.SYS on aina saatavilla. Sen jälkeen kopioidaan uusi CONFIG.SYS paikoilleen kirjoittamalla

```
COPY \SOFTA\CONFIG.EI \CONFIG.SYS
```

Järjestelmä saadaan mahdollisimman automaattiseksi vaihtamalla myös AUTOEXEC.BATia niin, että se käynnistää automaattisesti halutun sovelluksen. Kopioidaan siis myös alkuperäinen AUTOEXEC.BAT turvaan komennolla

```
COPY \AUTOEXEC.BAT \AUTOEXEC.VAR
```

ja kopioidaan sen tilalle uusi AUTOEXEC.BAT, joka on tallennettu nimelle AUTOEXEC.EI sovelluksen hakemistoon:

```
COPY \SOFTA\AUTOEXEC.EI \AUTOEXEC.BAT
```

Kun nämä valmistelut on tehty, kone käynnistetään uudelleen niksissä 64 esitetyllä WARMBOOT-ohjelmalla.

Nyt käytössä oleva AUTOEXEC.BAT rakennetaan niin, että siinä on vain sovelluksen käynnistämisen kannalta tarpeelliset komennot sekä tietenkin myös DOS-ympäristön tarvitsemat komennot, kuten näppäimistön ja mahdollisen PATHin asettaminen:

```
PATH C:\DOS;C:\SOFTA
KEYB SU,,C:\DOS\KEYBOARD.SYS
CD SOFTA
SOFTA
```

Kun sovelluksen käyttö sitten päättyy, komentojono jatkuu ja loput komennot palauttavat tilanteen takaisin lähtötilanteeseen:

```
COPY \CONFIG.VAR \CONFIG.SYS
COPY \AUTOEXEC.VAR \AUTOEXEC.BAT
WARMBOOT
```

Kun mikro tämän jälkeen käynnistyy, se on palannut alkuperäiseen tilaansa. Ohjelman omia aloitustiedostoja ei tarvitse kopioida talteen, sillä käyttäjä ei ole voinut muokata niitä sovelluksen käytön aikana. Jos muokkaus on tarpeen, tiedostoja on muokattava normaalissa DOS-tilassa.

Sovelluksen käynnistävä komentojono on kokonaisuudessaan seuraava:

```
ECHO Sovellus käynnistyy, odota...
COPY \CONFIG.SYS \CONFIG.VAR
COPY \SOFTA\CONFIG.EI \CONFIG.SYS
COPY \AUTOEXEC.BAT \AUTOEXEC.VAR
COPY \SOFTA\AUTOEXEC.EI \AUTOEXEC.BAT
WARMBOOT
```

Tarvittava AUTOEXEC.EI näyttää puolestaan seuraavalta:

```
PATH C:\DOS;C:\SOFTA
KEYB SU,,C:\DOS\KEYBOARD.SYS
CD SOFTA
REM Tähän sovelluksen käynnistyskomento
REM tai CALL xx.BAT, jos käynnistys tapahtuu
REM komentojonolla
SOFTA
REM Kun ajo on päättynyt, siivotaan
REM jäljet ja palataan takaisin...
COPY \CONFIG.VAR \CONFIG.SYS
COPY \AUTOEXEC.VAR \AUTOEXEC.BAT
WARMBOOT
```

Mikäli apuun otetaan vielä yksi AUTOEXEC.BAT-taso, päästään takaisin vaikkapa samaan käynnistysvalikkoon josta koko kierros lähti liikkeelle. Tämä AUTOEXEC.BAT asetetaan paikoilleen AUTOEXEC.EIn lopussa ja se palauttaa käyttäjän takaisin valikkoon. Koska tämä

komentojono on voimassa vain hetken, ennen sen loppumista pitää palauttaa alkuperäinen AUTOEXEC.VAR takaisin paikalleen.

Paperilla menettely näyttää monimutkaiselta, mutta käytännössä se toimii hyvin ja automaattisesti, kunhan A: asemassa vain ei ole levykettä. Jos asemassa on levyke, käynnistys tapahtuu siltä ja automatiikka katkeaa.

22

AUTOEXEC.BATin korvaaminen toisella komentojonolla

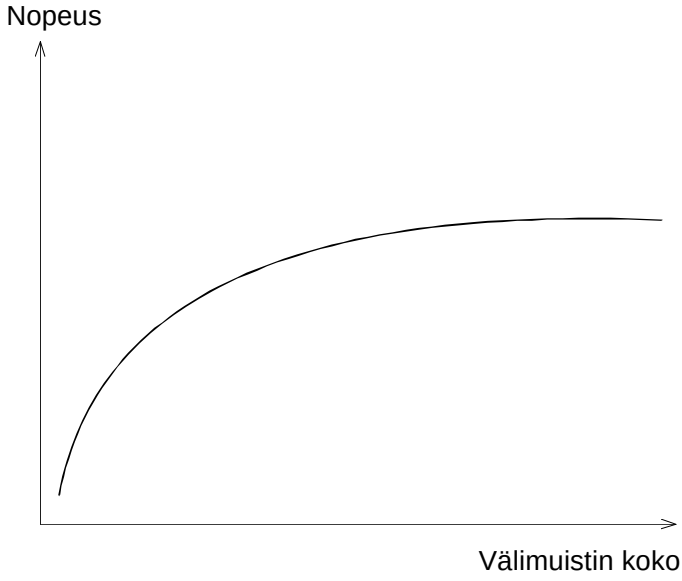
Paitsi sisäisten käskyjen muuttamiseen, COMMAND.COMin muokkausta voi käyttää myös aloituskomentojonon vaihtamiseen. Näin mikron asentaja voi varmistaa, että haluttu ohjelma tulee ajettua ensimmäiseksi kun kone käynnistetään eikä käyttäjä voi estää ajoa edes AUTOEXEC.BATia muokkaamalla.

AUTOEXEC.BATin kutsu etsitään niksissä 48 kuvatulla tavalla COMMAND.COMista ja sen tilalle kirjoitetaan vaikkapa OMAJONO .BAT. Koska nimi on lyhyempi kuin alkuperäinen, nimen ja pisteen väliin on lisättävä yksi välilyönti. Oikean kohdan löytäminen COMMAND.COMista pitäisi käydä helposti, sillä AUTOEXEC.BAT-tekstejä komentotulkissa on vain yksi.

Kun kone seuraavan kerran käynnistetään, se suorittaa OMAJONO. BATissa olevat komennot. Jotta myös perinteinen AUTOEXEC.BAT tulisi ajettua, sen kutsu lisätään OMAJONO.BATin loppuun. Kutsun eteen kannattaa lisätä kissanhäntämerkki, jotta niksi ei paljastuisi ruudulle tulostuvan tiedostonimen myötä.

Jos omassa käynnistysjonossa halutaan ajaa esimerkiksi salasanan kysyvä ohjelma, lopullinen OMAJONO.BAT näyttää seuraavalta:

```
SALASANA
@AUTOEXEC
```



Kun välimuistin koko kasvaa, levyaseman nopeus lisääntyy — mutta vain tiettyyn rajaan saakka ja siihenkin logaritmisesti. Ylisuuri välimuisti voi jopa hidastaa ohjelmia, varsinkin jos käytössä on kirjoituspuskuri.

23

Kuinka iso välimuisti kannattaa varata?

Mitä isompi välimuisti, sitä nopeammin monet levyä kuormittavat sovellukset toimivat. Nopeuden lisäys on kuitenkin logaritmista ja suurin suhteellinen lisäys saadaan jo pienellä välimuistilla. Muistin kasvattaminen 256:stä 512:een kilotavuun nopeuttaa levyä todennäköisesti yhtä paljon kuin kasvattaminen 512:sta 1536:een kilotavuun. Yli kahden megatavun kannattaa mennä tuskin koskaan, sillä jos välimuisti on liian iso, sen hallintaan alkaa kulua enemmän aikaa kuin mitä välimuistista saatava nopeushyöty on.

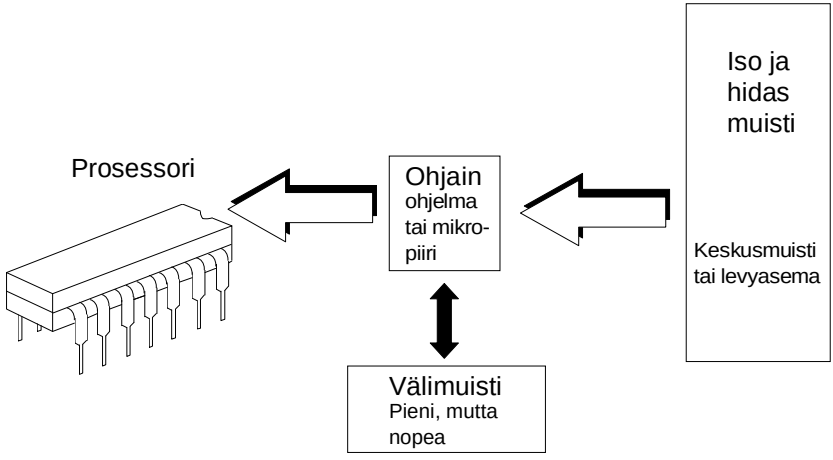
Erityisen tarkkaan välimuistin koko kannattaa harkita silloin, kun sovellukset pystyvät käyttämään jatkomuistia myös suoraan — kuten Windowsia käytettäessä. Tällöin kaikki välimuistille varattu jatkomuisti on pois Windowsilta ja hidastaa sitä, vaikka levytoiminnot nopeutuisivatkin.

Jatkomuistin lisääminen nopeuttaa Windowsin toimintaa ainakin neljään megatavuun asti. Sovelluksista riippuen muistin lisääminen saattaa nopeuttaa toimintaa vielä tästäkin eteenpäin, jopa kymmeneen megatavuun asti. Jos siis muistia on neljä megatavua tai vähemmän, välimuistille kannattaa varata enintään yksi megatavu. Isoilla, paljon keskusmuistia käyttävillä sovelluksilla puolikin megatavua saattaa antaa kokonaisuuden kannalta parhaan tuloksen.

Seuraavassa taulukossa on listattu WordPerfectin 5.1 Windows-versiolta käynnistykseen ja tiedoston avaukseen kulunut aika välimuistin koon funktiona. Testilaite oli 486SX 25 MHz ja siinä oli yhteensä neljä megatavua RAMia.

välimuisti (Mt)	aika (s)
0	24
0,5	18
1	23
1,5	29
2	37

Taulukosta nähdään, että ohjelma toimii nopeimmin kun neljän megatavun muistista on varattu välimuistiksi 512 kilotavua. Välimuistin tyypillä (luku/kirjoitus) ei ohjelman käynnistyksessä ja tiedoston avauksessa ole juurikaan vaikutusta. Sillä olisi kuitenkin vaikutusta myöhemmin WP:n normaalikäytössä, kun WP alkaa kirjoittaa levylle aputiedostoja tekstiä käsitellessään.



Sekä prosessorin että levyn välimuistit toimivat samalla periaatteella: pieni, nopea muistialue tehostaa suuren, hitaamman muistin käsittelyä pitämällä muistissaan aina viimeksi käsitellyt tiedot. Jos niihin viitataan myöhemmin uudestaan, tiedot löytyvät nopeammin.

24

Paljonko prosessorin välimuisti vaikuttaa?

Prosessorin välimuisti on aivan eri asia kuin kiintolevyn välimuisti, vaikka niistä käytetäänkin samaa nimeä ja englanninkielistä termiä cache. Tässä tapauksessa nimi viittaa toimintaperiaatteeseen eikä mihinkään yksittäiseen toteutukseen.

Sekä prosessorin että kiintolevyn välimuisti toimivat samalla periaatteella: pieni, nopea muistialue tehostaa isomman, hitaamman muistin käyttöä pitämällä omassa muistissaan viimeksi käsitellyt tiedot. Jos samoihin tai lähekkäin oleviin tietoihin viitataan hetken kuluttua uudelleen, tiedot löytyvät nopeasta välimuistista eikä hitaampaa isoa muistia tarvitse vaivata.

Toimintaperiaate prosessorin ja levyn välimuistissa on sama, mutta toteutus on aivan erilainen. Siinä, missä kiintolevyn välimuisti toteu-

tetaan yleensä ohjelmallisesti (kuten Smartdrv:llä), prosessorin välimuisti vaatii aina oman elektroniikan ja erikoisnopeita muistipiirejä. Prosessorin välimuisti nopeuttaa keskusmuistin (perus+jatkomuisti) käsittelyä, kun taas levyn välimuisti nopeuttaa levyaseman lukemista ja kirjoittamista (yleensä jatkomuistin avulla).

Intelin 486-prosessorit sisältävät kahdeksan kilotavun välimuistin, joka on rakennettu prosessorin sisään. Tämä välimuisti riittää tekemään 486-prosessoreista kaksi kertaa nopeampia kuin vastaavat 386-mallit. Eräissä IBM:n tekemissä prosessoriversioissa välimuistin koko on tuplattu 16 kiloon ja samaa määrää käytetään 586:ssa eli Pentiumissa.

Prosessorin sisäinen välimuisti on kaikkein tehokkainta, mutta se vie kallisarvoista tilaa piisirulla. Tästä syystä monet laitevalmistajat ovat lisänneet mikroihinsa ulkoisen välimuistin, joka vaatii paitsi välimuistiohjainpiirin myös muutaman erikoisnopean RAM-piirin. Ulkoisen välimuistin koko on tavallisesti 64 tai 256 kilotavua. Joissain mikroissa välimuistiohjain on valmiina, mutta nopeita RAM-piirejä ei ole. Jos käyttäjä haluaa nopeuttaa konettaan, hänen pitää ostaa välimuistin vaatimat muistipiirit itse.

Koska 386-sarjan prosessoreissa ei ole lainkaan sisäistä välimuistia, eräät valmistajat ovat lisänneet koneisiin ulkoisen välimuistin. Ulkoinen välimuisti ei kuitenkaan koskaan ole niin tehokasta kuin sisäinen eikä 386-prosessorin nopeuskaan ole niin suuri, että keskusmuistin lukeminen/kirjoittaminen ehtisi muodostua pullonkaulaksi. 486:sta alkaen tilanne on toinen.

Miten suuri merkitys ulkoisella välimuistilla sitten on? Jos koneessa on 64 kilon ulkoinen välimuisti, kannattaako se laajentaa 256 kiloon? Ja kannattaako ostohetkellä maksaa lisähintaa koneesta, jossa on ulkoinen välimuisti verrattuna koneeseen, jossa sitä ei ole?

Ulkoisen välimuistin merkitys on paljon pienempi kuin levyn välimuistin. Sen käyttö tuo vain vähän lisätehoa, sillä sisäinen välimuisti riittää tyydyttämään monet ohjelman tekemät muistiviittaukset. Monet sovellukset toimivat suurimman osan ajasta saman 16-bittisen segmentin sisällä, jolloin 64 kilon ulkoinen välimuisti riittää tyydyttämään useimmat muistiviittaukset.

Seuraava taulukko antaa karkean kuvan siitä lisätehosta, mikä saadaan välimuistia kasvattamalla kun käytetään merkkipohjaisia DOS-ohjelmia tai 16-bittisiä Windows-sovelluksia.

Välimuistin koon muutos	Tehonlisäys
0->64 kt	+10%
0->256 kt	+15%
64->256 kt	+4 %

Yli 64 kilon menevästä välimuistista saadaan täysi hyöty vasta sitten, kun sekä käyttöjärjestelmä että sovellukset ovat aidosti 32-bittisiä eikä 64 kilon segmenttirajoitus kummittele enää missään.

Siihen saakka välimuistin kasvattamiseen varatut markat kannattaa käyttää mieluummin muihin hankintoihin.

25

RAM-levyn avulla voi vähentää muistin määrää

RAM-levyä käytetään yleensä mikron nopeuttamiseen. Kun eniten käytetyt tiedostot kopioidaan RAM-levylle, niiden käyttö nopeutuu moninkertaiseksi kiintolevyyn verrattuna. Varjopuolena on tietenkin se, että RAM-levy tyhjenee heti kun sähkö katkeaa.

RAM-levyä voi käyttää myös toisin: se on näppärä tapa vähentää vapaaksi jäävää muistia. RAM-levylle varattu muisti on kiinteästi poissa Windowsilta ja OS/2:lta, joten vähentämällä RAM-levyn avulla näille käyttöjärjestelmille jäävää muistia voidaan tutkia, miten muistin määrä vaikuttaa niiden toimintaan ja mikä on pienin määrä muistia, jolla ne vielä käynnistyvät.

Varsinkin Windowsissa vapaan jatkomuistin määrä vaikuttaa selvästi ohjelmien toimintaan. Seuraavassa taulukossa on mitattu WordPerfect for Windowsin käynnistykseen ja dokumentin avaamiseen kulunut aika kun vaihteleva osa jatkomuistista oli poistettu käytöstä RAM-levyn avulla. Testikone oli 486SX 25 MHz ja välimuistille oli varattu 512 kilotavua.

RAM	aika (s)
2	86
2,5	39
3	31
3,5	20
4	18
4,5	16
5	16

Taulukko osoittaa selvästi, miten voimakkaasti ohjelma hidastuu kun vapaan muistin määrä vähenee. Taulukko osoittaa myös, että muistin kasvattaminen nopeuttaa WordPerfectiä aina 4,5 megatavuun asti.

Valitettavasti RAM-levystäkään ei ole apua silloin, kun käytetään lisäkortteja, jotka varaavat itselleen alueen jatkomuistista. Ainakin monet videokuvan kaappaamiseen tarkoitetut kortit ja myös eräät grafiikkakortit kuuluvat tähän ryhmään. Ne saattavat vaatia yhdestä neljään megatavua tilaa 16 megatavun alapuolelta, jolloin koneessa saa olla enintään 12-15 megatavua RAMia jotta kortit toimisivat. Jos muistia on enemmän, ainoa tapa saada kortti toimimaan on jatkomuistin määrän vähentäminen muistipiirejä poistamalla. RAM-levystä tai muista ohjelmallisista kikkakonsteista ei ole apua.

26

Ympäristömuuttujien tilan kasvattaminen

Tämä on klassinen kysymys, mutta otin sen mukaan kirjaan koska kysymys toistuu jatkuvasti: miten kasvatetaan ympäristömuuttujille varattua tilaa?

Ympäristömuuttujien tilan loppumisesta kertoo virheilmoitus

```
Out of environment space
```

Jos näin käy, ympäristömuuttujille varattua tilaa voidaan kasvattaa SHELL-asetuksella. Versiosta 3.2 alkaen asetus kirjoitetaan muodossa

```
SHELL=C:\COMMAND.COM /P /E:n
```

missä n on varattavien tavujen määrä. Oletusarvo on 192 (DOS 5.0:sta alkaen 256), mikä loppuu helposti kesken, sillä yhä useammat ohjelmat ovat alkaneet käyttää ympäristömuuttujien tarjoamia mahdollisuuksia. Varmuuden vuoksi voi kirjoittaa esimerkiksi asetuksen

```
SHELL=C:\COMMAND.COM /P /E:1488
```

joka varaa 1488 tavua.

Mikäli DOS-ohjelma toimii Windowsin alla, ympäristömuuttujien tilanvaraus määritellään SYSTEM.INI-tiedoston kohdassa [NonWindowsApps]. Esimerkiksi rivi

```
CommandEnvSize=1488
```

varaa jokaiselle käynnistyvälle DOS-ohjelmalle 1488 tavua. Tämä asetus on käytettävissä vasta Windows 3.1-versiosta lähtien. Vanhemmissa versioissa asia pitää hoitaa kiertotien kautta.

27

Paljonko ympäristömuuttujia on käytetty?

Mistä sitten näkee, kuinka paljon ympäristömuuttujille varatusta tilasta on käytetty ja paljonko on vielä vapaana? Esimerkiksi komentosarjalla

```
SET >TEMP.PTH
```

joka tulostaa kaikki määritellyt ympäristömuuttujat aputiedostoon. Sen pituus kertoo, kuinka paljon muuttujat vievät tilaa. Luku on hieman liian suuri, sillä tekstitiedostossa jokaisen rivin lopussa on kaksi merkkiä kun taas keskusmuistissa erotinmerkkinä on vain yksi tavu (nolla).

Pienellä kikkailulla kulutettujen ympäristömuuttujien määrä saadaan näytettyä ruudulla erikseen tai poimittua toiseen ympäristömuuttujaan. Varsinaisen työn tekee jono SELVITA.BAT, joka näyttää seuraavalta:

```
ECHO OFF
SET >APUXXXXX.TMP
DIR APUXXXXX.TMP|FIND "APUXXXXX TMP" >APUYYYYY.BAT
DEL APUXXXXX.TMP
APUYYYYY
```

Se suorittaa SET-komennon ja ohjaa tuloksen aputiedostoon, jonka nimitiedot poimitaan seuraavalla rivillä omaksi APUYYYYY.BAT-tiedostoksi. Kun tämä tiedosto käynnistetään ikään kuin se olisi aito komentojono, se kutsuu APUXXXXX.BAT-nimistä komentojonoa, joka näkee käytettyjen ympäristömuuttujien määrän toisena käynnistysparametrina. APUXXXXX.BAT pitää luoda etukäteen ja se voi näyttää esimerkiksi seuraavalta:

```
ECHO Muuttujia käytetty %2 tavua.
```

Jos halutaan, %2:n arvo voidaan sijoittaa omaan ympäristömuuttujaan vaikkapa komennolla

```
SET KAYTETTY=%2
```

mutta tällöin saatu arvo ei enää tarkkaan ottaen pidä paikkaansa, sillä KAYTETTY laskee vapaata tilaa.

28

Yli 640 kiloa perusmuistia

DOSissa on 640 kilon muistirajoitus, jota ei voi mitenkään kiertää.

Väärin.

DOSissa ei ole mitään 640 kilon rajoitusta, vaan se pystyisi käyttämään muistia aina yhteen megatavuun asti (tarkkaan ottaen 1114096 tavua jos HMA-aluekin lasketaan mukaan). Pahamaineinen

640 kilon muistirajoitus johtuu PC:n tekniikasta, joka varasi aikanaan 640-704 kilotavun (heksadesimaalisina A0000-AFFFF) osoitteet grafiikkamuistin käyttöön. Nykyään tällä alueella on grafiikkakortin kuvamuisti (EGA, VGA ja SVGA). Vanhat CGA- ja MDA-näytöt tarvitsevat vähemmän muistia ja niiden alueet sijoittuvat muistissa ylemmäksi, joten perusmuistia voitaisiin laajentaa MDA-tekstinäytöllä 704 kiloon ja CGA-näytöllä peräti 736 kiloon.

Tällaisia muistikortteja ei kuitenkaan löydy kaupoista, koska niillä olisi kovin vähän kysyntää — siksi harvalla on enää koneessaan CGA- tai MDA-näyttö. Onko sitten mitään tapaa laajentaa perusmuistia 640 kilosta eteenpäin jos koneessa on EGA- tai VGA-kortti?

On yksi, mutta se toimii vain merkkipohjaisilla ohjelmilla, eikä niilläkään kovin luotettavasti. Lisäksi se edellyttää, että koneessa on vähintään 386-prosessori.

Kaikki perustuu dynaamiseen muistinhallintaan, joka mahdollistaa muistin siirtämisen paikasta toiseen pelkillä ohjelmallisilla käskyillä. DOS 5.0:sta alkaen tämä tapahtuu EMM386.EXE-ohjelmalla. Vastaavia kaupallisia muistinhallintaohjelmia ovat mm. QEMM ja 386-Max.

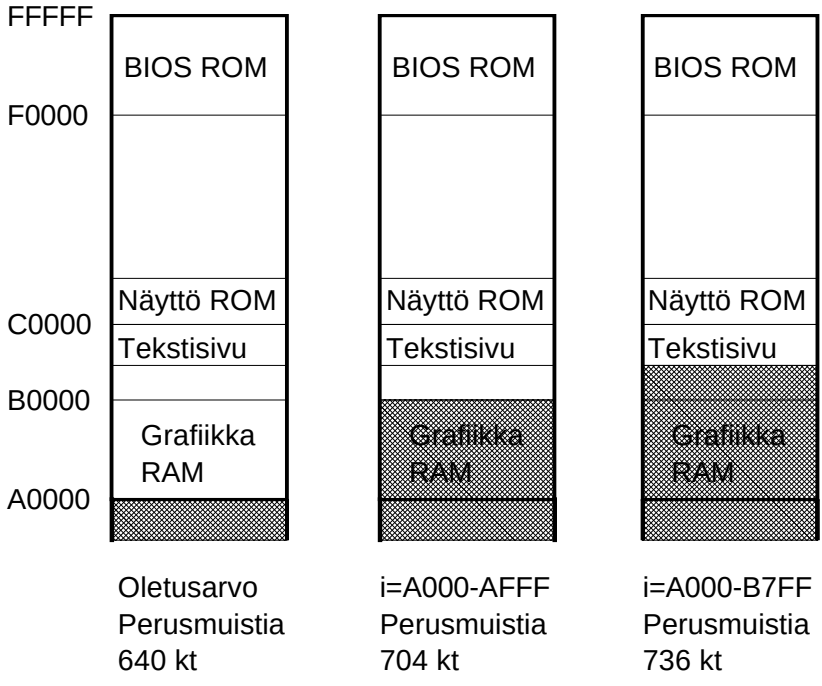
CONFIG.SYSiin tehty asetus

```
DEVICE=C:\DOS\EMM386.EXE i=A000-AFFF noems
```

siirtää palan jatkomuistista niin, että se peittää grafiikkakortin kuvamuistin. Sitä ei tarvita, kunhan tyydytään ajamaan vain pelkkiä tekstipohjaisia ohjelmia.

Kun muutos CONFIG.SYSiin on tehty, päästään katsomaan mitä se vaikuttaa käytännössä. Ainakin CHKDSK näyttää mielenkiintoisia muistilukemia:

```
Volume ASEMA D     created 27.12.1990 20.53
Volume Serial Number is 16C7-98F4

  34547712 bytes total disk space
   1748992 bytes in 4 hidden files
     96256 bytes in 37 directories
  30232576 bytes in 1005 user files
   2469888 bytes available on disk

      2048 bytes in each allocation unit
     16869 total allocation units on disk
```



EMM386-ohjain ja prosessorin dynaaminen muistinhallinta mahdollistavat perusmuistin kasvattamisen yli 640 kilon, mutta se edellyttää grafiikasta luopumista ja vähintään 386-prosessoria.

```
      1206 available allocation units on disk

    720896 total bytes memory
    695168 bytes free
```

Grafiikkamuistin jälkeen on yleensä 32 kilon tyhjä kohta (B0000-B7FFF) ennen näyttökortin tekstisivua ja ROM-alueita. Miksei siis saman tien otettaisi sekin käyttöön? Muutetaan CONFIG.SYS-riviä niin, että se peittää vielä enemmän ylämuistia:

```
DEVICE=C:\DOS\EMM386.EXE i=A000-B7FF noems
```

Nyt CHKDSK näyttää vielä hurjempia lukemia.

```
Volume ASEMA D     created 27.12.1990 20.53
Volume Serial Number is 16C7-98F4

  34547712 bytes total disk space
   1748992 bytes in 4 hidden files
     96256 bytes in 37 directories
  30236672 bytes in 1007 user files
   2465792 bytes available on disk

      2048 bytes in each allocation unit
     16869 total allocation units on disk
      1204 available allocation units on disk

    753664 total bytes memory
    727936 bytes free
```

Mitä sitten tapahtuu jos kaikesta huolimatta yritetään käyttää grafiikkaa? Mikro yrittää kirjoittaa kuvaruutumuistin alueelle, jolloin se kirjoittaa muistissa olevan komentotulkin yläosan päälle. Mikro lukkiintuu välittömästi ja näytöllä saattaa esiintyä erilaisia mielenkiintoisia efektejä. Kuten sanottua, grafiikkaa ei pidä edes yrittää!

Tästä niksistä voi olla hyötyä silloin, kun halutaan saada mahdollisimman paljon perusmuistia merkkipohjaisen sovelluksen käyttöön.

Ja ainakin sillä voi hämmästyttää kavereita.

29

Pilkusta piste

DOS 3.3:ssa KEYB-ohjelmaa muutettiin niin, että se alkoi Suomen tapauksessa tuottaa numeerisen piste-näppäimen sijaan pilkun. Microsoftin ohjelmoijat kuvittelivat varmaankin tehneensä suuren palveluksen suomalaisille PC-käyttäjille ja uskoivat helpottaneensa näiden elämää.

Vaan toisin kävi. Monet DOS-ohjelmat odottavat yhä vieläkin pistettä erottamaan luvun desimaaliosia toisistaan ja vain muutamissa ohjelmissa on mahdollista siirtyä käyttämään pilkkua, joka on toki virallinen ja standardinmukainen tapa. Windowsissa pilkku ei ole mikään ongelma, koska Windows osaa ottaa maakohtaiset erikoisuudet paremmin

huomioon ja oikein tehdyt Windows-sovellukset hyödyntävät niitä automaattisesti.

Mutta DOSissa ongelma on ja pysyy. Kirjoitusnäppäimistön pistettä on hankala käyttää, mikäli lukuja on paljon ja ne syötetään numeeriselta näppäimistöltä. Ongelmasta pääsee eroon vain muokkaamalla KEYBOARD.SYS-tiedostoa DEBUGilla niin, että se tuottaa pisteen pilkun sijaan. Ennen kuin KEYBOARD.SYSiä aletaan muokata, alkuperäinen tiedosto kannattaa kopioida levylle toiselle nimelle. Jos muokkauksessa sattuu virhe, alkuperäisen tiedoston kopio palautetaan takaisin KEYBOARD. SYSiksi.

Muutettava kohta vaihtelee DOS-versiosta riippuen. MS-DOS 5.0:ssa muutos tehdään avaamalla tiedosto käyttöön

```
C:\DOS>DEBUG KEYBOARD.SYS
```

ja tarkistamalla, mikä arvo löytyy tavusta 548B eteenpäin:

```
-d 548B
2C 00 00 0D 00
```

Jos arvo ei ole 2C, käytössä oleva KEYBOARD.SYS ei ole MS-DOS 5.0 ja muutososoite pitää etsiä erikseen. Jos arvo on 2C, se muutetaan 2E:ksi kirjoittamalla

```
-ecs:548B 2E
```

Tämän jälkeen tiedosto tallennetaan takaisin levylle

```
-w
```

ja DEBUGin käyttö lopetetaan.

```
-q
```

Kun mikro käynnistetään uudelleen, numeronäppäimistön pisteestä pitäisi tulostua piste.

Periaate muissa DOS-versioissa on sama, mutta osoite vaihtelee. Esimerkiksi PC-DOS 5.0:ssa osoite on 550B ja 3.3:ssa 450B. Oikean osoitteen voi selvittää etsimällä Debugin S-komennolla (tai suoraan tiedostosta Nortonin tai PC Toolsin avulla) tiedostosta kohtaa, jossa

esiintyvät tavut 2C 00 00 0D 00. Yleensä tiedostosta löytyy useita tällaisia tavujonoja, jolloin kokeilu kannattaa aloittaa viimeistä alkaen. Debugia käytettäessä kannattaa ensin laskea KEYBOARD.SYS-tiedoston pituus heksadesimaalisena ja käyttää sitä S-komennon ylärajana (eli -s 0000 xxxx 2C 00 00 0D 00 missä xxxx on tiedoston laskettu pituus heksana).

Ellei oikeata osoitetta löydy, voi käyttää jotakin muistinvaraista shareware-apuohjelmaa joka tekee saman. Eräs tällainen ohjelma kulkee nimellä COMMA.ZIP.

30

Poista turhat viittaukset PATHista

PATH-komennolla kerrotaan, mistä hakemistoista DOSin pitää etsiä käynnistettävää ohjelmatiedostoa, jos sitä ei löydy oletushakemistosta. Tarkoitus on, että PATH osoittaisi yleisesti tarvittavia apuohjelmia. Sovellusten hakemistoja ei pitäisi kirjoittaa PATHiin kuin poikkeustapauksissa.

Koska PATH-komennon enimmäispituus on vain 127 merkkiä, polkuun ei kannata määritellä yhtään turhaa hakemistoa. Paitsi että turhat viittaukset vievät tilaa, ne myös hidastavat toimintaa, koska DOS joutuu käymään jokaisen merkityn hakemiston läpi.

Toisin kuin Unixissa, DOS etsii käynnistettävää ohjelmaa aina ensiksi oletushakemistosta. PATHiin on siten turha laittaa Unixin tapaista pistettä.

Vielä yleisempi tapa on kirjoittaa PATHiin viittaus C: aseman päähakemistoon. Usein tämä viittaus on vieläpä PATH-rivin alussa:

```
PATH C:\;C:\DOS;C:\UTIL;C:\BAT;C:\UTIL\NORTON
```

Päähakemiston listaaminen PATHissa on täysin turhaa, koska päähakemistossa ei pitäisi olla lainkaan ajettavia ohjelmia.

30 B

Jos PATHin pituus kaikesta huolimatta loppuu kesken, sitä ei voi mitenkään helposti kasvattaa. On olemassa joitakin sharewareapuohjelmia (kuten XPATH), jotka laajentavat polun pituutta, mutta rajoitus on niin syvällä DOSissa että laajennetut PATHit eivät välttämättä toimi kaikissa tilanteissa oikein.

DOS tarjoaa itse vain yhden tavan PATHin pidentämiseen ja se perustuu SUBSTin käyttöön. Luomalla SUBSTin avulla näennäinen levyasema ja ohjaamalla PATH sen kautta saadaan PATH-listauksen pituutta lyhennettyä.

Esimerkiksi PATH, joka näyttää seuraavalta

```
PATH C:\DOS;E:\UTIL\MUUT;D:\SOFTAT\PD
```

voidaan muuttaa SUBSTin ansiosta muotoon

```
PATH C:\DOS;X:\;Y:\
```

kunhan näennäiset asemat luodaan ensin kirjoittamalla

```
SUBST X: E:\UTIL\MUUT
SUBST Y: D:\SOFTAT\PD
```

Kenoviivat asemien nimien jälkeen voidaan jättää pois, jolloin PATHin pituus lyhenee entisestään. SUBST-komennon käyttö edellyttää, että CONFIG.SYSissä oleva LASTDRIVE-asetus on vähintään sama kuin viimeinen SUBSTilla käytetty asemakirjain.

SUBSTilla luodut ja PATHin kautta viitatut asemat tuottivat ongelmia vielä Windows 3.0:ssa, sillä Windows ei käynyt tutkimassa niitä. Versiosta 3.1 alkaen asia on kuitenkin korjattu joten SUBSTia voidaan käyttää. Varmuuden vuoksi SUBST-komentoja ei kuitenkaan tule antaa Windowsin DOS-ikkunassa vaan tarpeelliset määritykset pitää tehdä jo ennen Windowsin käynnistystä. Jos SUBST-asemat halutaan poistaa, ne saa poistaa vasta kun Windowsin ajo on lopetettu.

31

APPEND on yleensä tarpeeton

DOS 3.3:n myötä tullut APPEND-apuohjelma laajentaa PATHin toimintaa niin, että ohjelmatiedostojen lisäksi voidaan määritellä erityinen työtiedostojen hakupolku. Valitettavasti APPENDin käyttö tuottaa erilaisia sivuilmiöitä ja saattaa häiritä DOSin omien apuohjelmien, kuten BACKUPin toimintaa. Koska APPEND kuluttaa myös keskusmuistia, se kannattaa poistaa AUTOEXEC.BATista, jos asennusohjelma on sen mennyt sinne lisäämään.

Tilanteet, joissa APPENDia todella tarvittaisiin, ovat harvinaisia.

32

ANSI hidastaa ruudulle tulostamista

ANSIa tarvitaan niksissä 51 esitettyihin kuvaruudun värimäärityksiin, mutta sitä tarvitsevia sovellusohjelmia on markkinoilla yhden käden sormin lueteltava määrä. Koska ANSI-ohjain on käytännössä tarpeeton ja koska se kuluttaa muutaman kilotavun kallisarvoista keskusmuistia, se kannattaa yleensä poistaa.

Muistin kuluttamisen lisäksi ANSI-ohjain hidastaa ruudulle tapahtuvaa tulostusta, mikä näkyy selvästi esimerkiksi DIR- ja TYPE-käskyjen kohdalla. Kun ANSI on ladattu muistiin, se tarkkailee koko ajan ruudulle tulostuvaa tekstiä ja tutkii, löytyykö tulostuvasta tekstivirrasta sille itselleen tarkoitettuja ohjauskoodeja. ANSI ei kuitenkaan vaikuta ohjelmiin, jotka kirjoittavat suoraan kuvaruutumuistiin. Kaupallisista sovelluksista suurin osa on tällaisia.

Kokeilin ANSIn hidastavaa vaikutusta kahdella mikrollani. Testiaineistona oli 510 kilotavun tekstitiedosto, jonka tulostin kuvaruudulle TYPE-käskyllä. Ensimmäinen testikone oli vanha 16 megahertsin 386, jossa oli hidas 8-bittinen perus-VGA. Toinen kone oli 33 megahertsin

486-klooni, jossa oli 16-bittinen näytönohjain. Tulokset olivat seuraavan taulukon mukaisia:

Mikro	Ilman ANSIa	ANSIn kanssa	Hidastus
386/16	434 s	693 s	59,7 %
486/33	78 s	100 s	28,2 %

Paitsi että ANSI hidastaa kuvaruudulle tulostamista, se on myös tietoturvariski. ANSI-koodeilla voidaan nimittäin tehdä määrityksiä, jotka muuttavat näppäinten merkityksiä vaikkapa niin, että A-kirjaimesta tuleekin DEL-käsky. Määrityksessä tarvittavat ANSI-koodit voidaan upottaa tekstitiedostoon jolloin pelkkä tiedoston tulostaminen TYPE-komennolla riittää aktivoimaan siinä olevat komennot.

ANSI-ohjain kannattaa siis jättää lataamatta monestakin syystä.

33

ANSIa tarvitsevat ohjelmat

Mistä sitten tietää, käyttääkö ohjelma ANSI-ohjauskoodeja? Tällaiset ohjelmat on helppo tunnistaa siitä, että ANSIn puuttuessa niiden kuvaruudulle tekemä tulostus on sekaisin ja vilisee hakasulkuja. Sulkujen jäljessä näkyy numeroita, jotka ovat kohdistimen ohjaukseen tarkoitettuja koordinaatteja. Kun ANSI on ladattu, se nappaa koodit itselleen ja toimii niiden ohjauksen mukaisesti. Jos ohjainta ei ole ladattu, koodit "vuotavat" näytölle ja sotkevat sen.

Myös eräät DOSin omat toiminnot käyttävät ANSIa. Esimerkiksi MODE osaa vaihtaa näyttötilan 43:lle tai 50:lle riville vain, jos ANSI on käytössä. Muutoin tulostuu virheilmoitus:

```
C:\>MODE con lines=43

ANSI.SYS must be installed to perform requested
function
```

34

Kiintolevyltä toiselle

Tiedostojen siirto kiintolevyltä toiselle on väistämättä työläs operaatio. Jos vain mahdollista, työ kannattaa tehdä tiedonsiirto-ohjelmalla. Koneiden sarja- tai rinnakkaisportit yhdistetään kaapelilla, jonka jälkeen tiedonsiirto-ohjelma (kuten Laplink) lähettää bitit koneesta toiseen suurella nopeudella.

Mutta jos ohjelmaa ja sen vaatimaa kaapelia ei ole käytettävissä, on turvauduttava muihin konsteihin. Mikäli molemmissa koneissa on samantyyppinen levyasema (IDE, SCSI, ESDI), asemat voi ehkä kytkeä siirron ajaksi samaan koneeseen. Tämä tosin vaatii koneiden avaamisen, asemien irrottamisen ja SETUP-tyypin määräämisen kiinnitettävälle levylle, mutta kun ne on tehty, varsinainen tiedostojen kopiointi käy nopeasti.

Jos tämäkään ei toimi, jää ainoaksi mahdollisuudeksi siirtää tiedostot levykkeillä koneesta toiseen. Tehdään ensimmäisen koneen tiedostoista BACKUP ja palautetaan se toisen koneen RESTORElla. Ja ennen siirtoa kannattaa varmistaa, että molempien koneiden DOS-versiot ovat samat jotta toisen koneen RESTORE pystyy varmasti lukemaan ensimmäisen koneen BACKUPin tuottamia levykkeitä. Yksinkertaista, mutta työlästä.

Jos kiintolevyn kapasiteetti on 80 megatavua, levy on täynnä ja molemmissa koneissa on 1,44 megatavun korput niin mikä on pienin korppujen määrä, jolla tiedostojen siirto-operaatiosta selvitään?

Vastaus: yksi.

Ensimmäisessä koneessa käynnistetään BACKUP ja toisessa RESTORE. Levyke laitetaan asemaan kun BACKUP pyytää sitä. BACKUP kirjoittaa levykkeelle aikansa ja pyytää sen jälkeen toista levykettä.

Silloin otetaan levyke ja siirretään se toiseen koneeseen, jossa RESTORE odottaa jo innokkaasti ensimmäistä levykettään. Sille annetaan BACKUPin tuottama aineisto, jonka se palauttaa kiintolevylle ja pyytää sen jälkeen kakkoslevykettä. Tällöin levyke siirretään takaisin ensimmäiseen koneeseen, jolloin BACKUP kuvittelee sitä levykkeeksi numero kaksi. Levykkeen täytyttyä kakkoslevyke annetaan odottavalle

RESTORElle ja tätä toistetaan, kunnes kaikki tiedostot on saatu siirrettyä.

Kahdella levykkeellä työ käy vielä nopeammin, koska ensimmäisen mikron BACKUP voi kirjoittaa toiselle levykkeelle samalla kun toisen mikron RESTORE lukee edellistä levykettä.

Yhden tai kahden levykkeen käyttö on nopeampaa kuin täyden varmistuksen ja palautuksen suorittaminen, sillä sekä varmistus että palautus voivat olla käynnissä yhtä aikaa. Lisäksi se säästää levykkeitä, joita muuten pitäisi varata kokonainen pino. Kun molemmat koneet sijoittaa lähekkäin, levykkeen siirtäminen koneesta toiseen käy näppärästi.

Menetelmän ainoa varjopuoli on siinä, että jos siirrossa sattuu jokin hama, koko prosessi pitää aloittaa alusta.

35

FDISK ja kiintolevyn liialliset urat

Jotta mikro pystyisi käyttämään kiintolevyä, sen pitää tietää kiintolevyn tekniset parametrit. Näitä tietoja säilytetään paristolla tai akulla varmistetussa CMOS-muistissa ja tietoja päivitetään tarvittaessa SETUP-ohjelmalla. Kiintolevyyn viitataan tyyppinumerolla, jonka jälkeen mikro osaa käyttää levyä.

Ongelmia syntyy silloin, kun koneeseen ostetaan uusi kiintolevy eikä SETUPin tuntemista tyypeistä mikään täsmää täysin levyn kanssa. Tällöin voidaan menetellä niin, että valitaan tyyppinumero jossa on enemmän uria (sylintereitä) kuin levyllä todellisuudessa, mutta jonka muut parametrit (kuten levypuolten määrä sekä mahdolliset lukupään laskeutumis- ja magneettikentän vahvistusurat) täsmäävät levyn kanssa. Tämän jälkeen FDISKillä määritellään osiot siten, että vain levyllä todellisuudessa olevat sylinterit otetaan käyttöön.

Esimerkki valaisee parhaiten tilannetta. Oletetaan, että levyllä on vanhaan tapaan 17 sektoria, 4 puolta ja 800 uraa. Parhaiten vastaava SETUP-tyyppi sisältää kuitenkin 900 uraa, missä on 100 uraa liikaa. Yhdelle uralle mahtuu 17*512 eli 8,5 kilotavua. Sylinteriin kuuluvat

kaikki neljä levynpuolta joten yhden sylinterin koko on 34 kilotavua ja sadan sylinterin vastaavasti 3400 kilotavua eli noin 3,4 megatavua.

Kun koko levyn kapasiteetti on 900*17*4*512 tavua eli 31334400 tavua (siis 29,88 megatavua), määritellään ensiöosion kooksi 29,88-3,4 = 26,48 megatavua. Loppu 3,4 megatavua jää kuvitteellisen jatketun osion alueelle, eikä sitä yritetäkään käyttää.

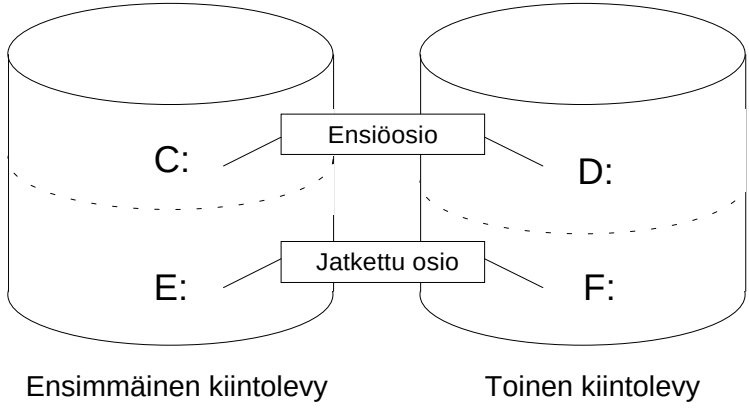
Jos levy on iso, jatkettu osio voidaan perustaa normaaliin tapaan kunhan huolehditaan siitä, että sen sisältämän viimeisen loogisen levyasematunnuksen koko on 3,4 megatavua.

Vanhoilla DOS-versioilla (3.3 ja vanhemmat) esimerkin kaltaiset määritykset ovat erityisen helppoja, koska niissä osion koot määritellään megatavujen sijaan muutenkin sylintereinä.

36

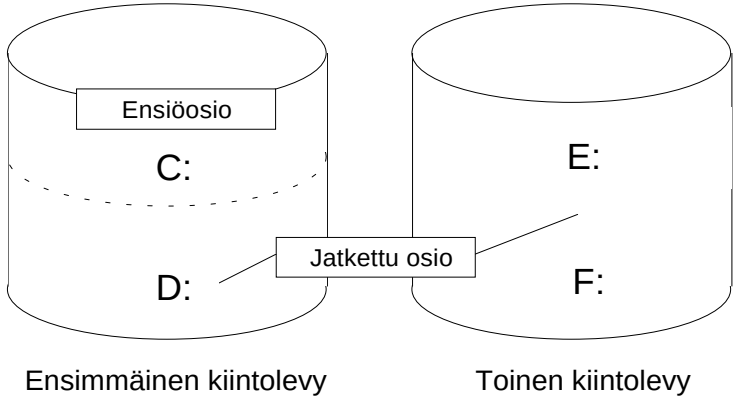
FDISK ja toisen kiintolevyn ensiöosio

Yhä useampi joutuu hankkimaan koneeseensa toisen kiintolevyn, kun ensimmäinen on käynyt liian pieneksi. Kun DOS numeroi levyjä, se antaa ensimmäisen fyysisen levyn ensiöosiolle tunnuksen C: mutta hyppää sen jälkeen toiselle levylle niin, että sen ensiöosio saa kirjaimen D. Ensimmäisen levyn jatketussa osiossa olevat loogiset levyasemat tulevat vasta tämän jälkeen.



Jos ensimmäinen levyasema on aiemmin ollut jaettu C: ja D: tunnuksiin, niistä tulee toisen levyn asentamisen jälkeen C: ja E: mikä riittää sotkemaan ohjelmiin tehdyt asetukset ja pakottaa muuttamaan esimerkiksi Windowsin Program Managerin polkumäärityksiä.

Tältä ongelmalta voi välttyä siten, ettei perusta toiselle kiintolevylle lainkaan ensiöosiota. Sitä ei tarvita, koska toiselta levyltä ei kuitenkaan voisi käynnistää konetta. Kun toiselle levylle perustetaan pelkkä jatkettu osio ja se jaetaan loogisiin tunnuksiin, nämä tunnukset sijoittuvat kaikkien ensimmäisellä asemalla olevien tunnusten jälkeen eikä vanha asemajärjestys häiriinny:



37

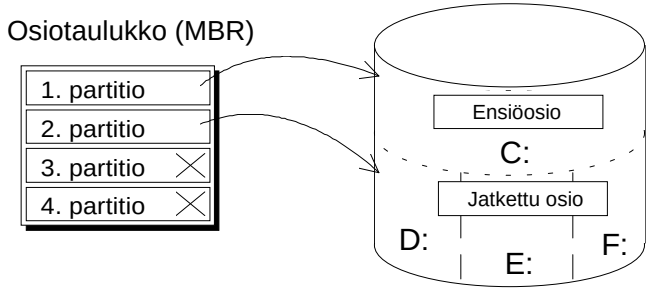
Partitio ei ole sama asia kuin looginen levyasema

"Levyni on jaettu kahteen partitioon: C ja D."

"Asennan ohjelman E partitioon."

Tuttuja lauseita monen käyttäjän ja jopa mikrotukihenkilön suusta. Mutta vääriä. Ne osoittavat, ettei puhuja ole ymmärtänyt partition merkitystä oikein.

Mikä siis on partitio?



Osiotaulukko kertoo, miten partitiot on jaettu. DOS voi käyttää neljästä partitiopaikasta enintään kaksi. Ensiöosio näkyy C: asemana ja jatkettu osio on jaettava loogisiin asematunnuksiin, jotka saavat kirjaimet D:stä alkaen. Normaalikäytössä osiotaulukon kaksi viimeistä riviä ovat tyhjiä.

Historiallisista ajoista lähtien PC:n kiintolevy on voitu jakaa neljään täysin erilliseen alueeseen. Nämä alueet on eristetty toisistaan niin täydellisesti, että niillä voi olla jopa eri käyttöjärjestelmät eikä ole pelkoa siitä, että tiedostot menisivät päällekkäin.

Tieto siitä, miten moneen osaan (osioon, partitioon) levy on jaettu ja mistä eri osat alkavat, tallennetaan FDISK-ohjelmalla osiotaulukkoon (MBR, Master Boot Record). Taulukon lisäksi MBR sisältää myös pienen latausohjelman, joka on tärkeä viruksia torjuttaessa.

Osiotaulukossa on neljä riviä eli tilaa neljän eri partition tiedoille. Yleensä riveistä kaksi tai kolmekin on täysin tyhjiä, koska levylle on määritelty vain yksi tai kaksi partitiota. Versioon 3.2 saakka DOS pystyy käyttämään vain yhtä osiota. Seuraavassa versiossa DOSia paranneltiin niin, että käyttöön voitiin ottaa myös toinen osio. Tätä jatkettua osiota (extended partition) ei kuitenkaan voitu käyttää sellaisenaan vaan se piti edelleen jakaa loogisiin levyasemiin, jotka sitten näkyivät käyttäjälle kirjaimilla D, E, F ja niin edelleen.

Versiosta 4.0 lähtien tuli mahdolliseksi käyttää isoja levyasemia yhtenä osiona ilman, että jakoa loogisiin levyasemiin oli pakko tehdä. Jos käyttöön valittiin vain yksi osio, koko levy näkyi yhtenä isona C: asemana. Mikäli levy haluttiin jakaa pienempiin osiin, piti avuksi jälleen ottaa jatkettu osio ja perustaa sinne loogisia levyasemia.

Käyttäjän kannalta sekaannus tapahtuu siinä, ettei tiedetä mitä eroa on loogisella levyasemalla ja partitiolla.

Siksi kertaus: levyllä voi olla enintään neljä osiota, joista DOS pystyy käyttämään enintään kahta. Osioista ensimmäinen näkyy C: asemana, mutta toinen osio pitää jakaa ennen käyttöä loogisiin levyasemiin. Olipa asemia miten monta tahansa, ne kaikki sisältyvät toiseen partitioon. Jos asemia perustetaan vain yksi, se näkyy D:nä ja kattaa koko jatketun osion alueen.

38

Älä laita kaikkia munia samaan koriin

DOS 4.0:sta lähtien on ollut mahdollista käyttää isoakin levyasema yhtenä C: asemana, kun ehdotonta pakkoa pienempien loogisten levyasemien käyttöön ei enää ole ollut. Jako pienempiin asemiin kannattaa silti yhä, koska jokaisella loogisella levyasemalla on oma, itsenäinen kirjanpitonsa. Vaikka C: aseman kirjanpito tuhoutuisi tai käyttäjä alustaisi sen vahingossa, muilla asemilla olevat tiedostot säästyisivät.

Turvallisuuden lisäksi pienet loogiset asemat ovat myös taloudellisempia, koska tiettyjen raja-arvojen jälkeen ne siirtyvät käyttämään pienempää varausyksikköä, mikä puolestaan merkitsee pienempää hukkatilaa. Nämä varausyksikön kokoon liittyvät raja-arvot on esitelty niksissä 85.

Pienten asematunnusten ainoa haittapuoli on niiden vaikeampi hallittavuus. Koska asemien rajat ovat kiinteitä, yksi asema saattaa tulla täyteen vaikka muilla olisi vielä runsaasti tilaa.

38 B

Olipa osiojako tehty miten tahansa, sen tuloksena syntynyt osiotaulukko kannattaa varmistaa levykkeelle heti osioiden määrityksen jälkeen. DOS 5.0:sta alkaen tämä tapahtuu komennolla MIRROR /PARTN. Se pyytää levykettä, jolle osiotaulukon kopio kirjoitetaan. Koska osiotaulukko voi

muuttua vain FDISK-ohjelman käytön jälkeen, varmistusta ei tarvitse tehdä kuin kerran.

Jos osiotaulukko jostain syystä sekoaa, DOS ei enää tunnista koko levyasemaa. Kun levykkeelle varmistettu taulukko palautetaan takaisin kiintolevylle komennolla UNFORMAT /PARTN, päästään ainakin tutkimaan ovatko muut kirjanpitoalueet säilyneet kunnossa.

Levykkeelle tallennetusta osiotaulukosta on apua myös silloin, kun halutaan tappaa osiotaulukkoon pesiytynyt virus. Palauttamalla alkuperäinen taulukko saastuneen päälle saadaan virus tapettua.

39

Kiintolevyn parametrien näyttäminen

DOS 5.0:sta alkaen käyttöjärjestelmän mukana toimitetaan UNFORMAT-ohjelma, joka on lisenssioitu Central Pointilta. Saman ohjelman vanhempia versioita on toimitettu myös PC Toolsin mukana.

UNFORMAT on tarkoitettu tilanteeseen, jossa levy on vahingossa alustettu. UNFORMAT etsii levyltä kirjanpitoalueista viimeksi tehdyn kopion ja palauttaa sen takaisin käyttöön. Ohjelmassa on myös hyödyllinen, mutta vähän tunnettu ominaisuus, joka saa sen näyttämään kiintolevyn tekniset parametrit: levypuolten, sektorien ja sylinterien määrän. Tämä tehdään komennolla

```
C:\>UNFORMAT /PARTN /L

Hard Disk Partition Table display.
Drive # 80h has 822 cylinders, 10 heads, 17 sectors
(from BIOS).
```

Kun muistetaan, että yhteen lohkoon (sector) mahtuu 512 tavua, voidaan laskea kiintolevyn kapasiteetti: 822 x 10 x 17 x 512 = 71546880 eli 68,23 megatavua.

Tietojen jälkeen UNFORMAT-ohjelma tulostaa myös osiotietojen määritykset. Ne ovat samat kuin FDISKin näyttämät, mutta selkeämmässä muodossa.

40

KISS-periaate

Keep It Simple, Sweethart! Eli: pidä se yksinkertaisena, kultaseni.

Aloittelevan mikron käyttäjän tuntee siitä, että hänen CONFIG.SYSinsä ja AUTOEXEC.BATinsa ovat lyhyet ja yksinkertaiset. Aloittelija kun ei tiedä (eikä välitä), mitä kaikkea niihin voisi kirjoittaa.

Harrastelijan tuntee siitä, että hänen CONFIG.SYSinsä ja AUTOEXEC.BATinsa ovat pitkiä kuin assembler-listaukset. CONFIG.SYSissä on asetettu jokainen kuviteltavissa oleva vipu johonkin asentoon ja AUTOEXEC.BAT lataa muistiin kaikki mahdolliset ja pari mahdotontakin muistinvaraista apuohjelmaa.

Ammattilaisen tuntee siitä, että hän on näennäisesti palannut takaisin lähtöpisteeseen. Hän on jo oppinut, että yksinkertaisuus on kauneutta — ja mikä vielä tärkeämpää: varmuutta. Hän poistaa muistista kaikki ylimääräiset laiteohjaimet ja muistinvaraiset ohjelmat, hän käyttää säästeliäästi kaupallisia apuohjelmia ja tyytyy mielellään DOSin mukana tuleviin yksinkertaisiin ratkaisuihin.

Kenen laitteisto toimii parhaiten?

Aloittelija ei tiedä, mitä hänen pitäisi koneeltaan vaatia. Hän on tyytyväinen, kun skandit toimivat ja tiedostojen tekoajat näkyvät oikeassa muodossa.

Harrastelija saa iloa viritellessään konettaan, kokeillessaan uusia apuohjelmia, kehittäessään riemunkirjavia prompteja sekä järjestäessään kiintolevyn tiedostot säännöllisesti peräkkäin unfragmentointiohjelmalla.

Ammattilaisen kone toimii parhaiten. Se jättää eniten muistia vapaaksi ja toimii kaikista luotettavimmin, koska turhat lisäykset on karsittu eivätkä ne silloin voi myöskään tuottaa ongelmia. Siinä ei ehkä ole kaikkia niitä hienouksia mitä harrastelijan koneessa on, mutta ammattilainen ei kaipaakaan niitä. Koneessa on vähän, mutta hyvää. Eikä ammattilainen maksa turhista apuohjelmista.

Tämän niksin ydin on seuraava: vältä kaikkia virityksiä! Loppujen lopuksi ne tuottavat enemmän harmia kuin hyötyä.

41

Dokumentoimaton COMMENT-asetus

DOS 4.0:sta lähtien CONFIG.SYSissä voidaan käyttää dokumentoimatonta COMMENT-lausetta. Sen perään kirjoitetaan kaksi merkkiä, joilla alkavat rivit ohitetaan ikään kuin ne olisivat kommentteja.

Esimerkiksi asetus

```
COMMENT=LA
```

ohittaa kaikki LA-alkuiset rivit, kuten LASTDRIVE=Z.

On hieman vaikeata nähdä, mihin käyttöön DOS 4:n tehneet IBM:n suunnittelijat ovat tätä komentoa ajatelleet. Sitä voisi kenties käyttää muiden asennusohjelmien huijaamiseen niin, että ne näkevät CONFIG.SYSissä tarvitsemansa rivit, jotka eivät kuitenkaan vaikuta mitään.

Esimerkin LA-määrityksen jälkeen asennusohjelma luulee, että LASTDRIVE on määritelty Z:ksi eikä ala muuttaa sitä, vaikka asetuksella ei COMMENTin vuoksi olekaan mitään vaikutusta.

42

Ohjelman asentaminen B: asemasta

Kaikki ohjelmoijat eivät vieläkään suostu uskomaan, että käyttäjillä saattaisi olla kaksi levykeasemaa. Itsepintaisesti he kirjoittavat asennusohjelmia, jotka olettavat levykkeen olevan A: asemassa eivätkä anna mahdollisuutta aseman muuttamiseen.

Jos asennusohjelma on tavallinen komentoriviltä käynnistettävä DOS-sovellus sen huijaaminen on helppoa: ASSIGN-komennolla vaihdetaan B: aseman kirjaimeksi A: jonka jälkeen asennus käynnistetään normaalisti. Työn jälkeen asemat palautetaan entiselleen kirjoittamalla pelkkä ASSIGN.

```
ASSIGN A: B:
```

<asennus>

```
ASSIGN
```

Aina tämä ei kuitenkaan auta. Jos asennus edellyttää koneen käynnistämistä alkuperäisiltä levykkeiltä (esimerkiksi OS/2 tai vanhat DOS-versiot), ASSIGN-niksiä ei voida käyttää. Ellei koneen BIOSissa ole mahdollisuutta ohjata käynnistystä B: asemalle, ei asiaan pysty vaikuttamaan ohjelmallisin keinoin.

Silloin jää ainoaksi mahdollisuudeksi vaihtaa A: ja B: asemiin menevät kaapelit ristiin ja kertoa SETUPin avulla BIOSille, että korppuaseman tilalla onkin nyt lerppu ja päinvastoin. Asennuksen jälkeen pitää vielä muistaa palauttaa johdot takaisin oikein päin.

43

Jatkomuistin käsittely vie aikaa

Kun DOS-ohjelmat käyttävät jatkomuistia, prosessori joudutaan siirtämään hetkeksi suojattuun tilaan (protected mode) jolloin keskeytykset estyvät. Tästä syystä DOS-ohjelmat voivat käyttää jatkomuistia vain pienen hetken kerrallaan. Aina välillä niiden on palattava takaisin prosessorin vapaaseen (real mode) -tilaan, jotta keskeytykset eivät hukkuisi kokonaan.

Keskeytysten estyminen lyhyeksikin aikaa kerrallaan saattaa tuottaa ongelmia sellaisten ohjelmien kanssa, jotka tarvitsevat ahkerasti keskeytysten palveluita. Tällaisia ovat esimerkiksi Laplinkin kaltaiset tiedonsiirto-ohjelmat. Ne saattavat antaa siirtovirheitä, mikäli niitä käytetään yhdessä jatkomuistissa toimivan RAM-levyn tai välimuistin kanssa. Jotta tiedostojen siirto voisi tapahtua, jatkomuistia käyttävät ohjelmat pitää poistaa siirron ajaksi.

Joskus pelkkä parametrin muuttaminen saattaa auttaa. Esimerkiksi DOSin omassa VDISKissä jatkomuistin puolella vietettävää aikaa voidaan rajoittaa /E: valitsimella. Sen perään kirjoitetaan kerralla siirrettävien lohkojen määrä (1-8). Valitsemalla arvo riittävän pieneksi

saadaan tiedonsiirto toimimaan oikein. Vastaava asetus löytyy myös joistakin välimuistiohjelmista (kuten Hcache), mutta DOSin ja Windowsin mukana tulevissa Smartdrv-versioissa sitä ei ole katsottu tarpeelliseksi.

44

SMARTDRV:n ansat

Kun käytetään komentoriviltä asennettavaa Smartdrv-välimuistia (versio 4.0 tai uudempi) on tärkeätä, että se asennetaan muistiin vasta KEYB-näppäimistöohjelman jälkeen. Tällöin Smartdrv pystyy valvomaan näppäimistöä ja kun käyttäjä painaa Ctrl+Alt+Del käynnistääkseen koneen uudelleen, Smartdrv tutkii onko muistipuskureissa vielä tietoa, jota ei ole ehditty kirjoittaa levylle.

Jos tällaista tietoa on, Smartdrv tulostaa ruudun vasempaan ylänurkkaan laatikon, jossa lukee "Waiting for shutdown" ja suorittaa tiedostojen kirjoittamisen loppuun asti. Vasta tämän jälkeen se antaa uudelleenkäynnistyksen tapahtua.

Jos asennus on tehty toisinpäin, Smartdrv ei huomaa Ctrl+Alt+Deliä ja uudelleenkäynnistys tapahtuu heti. Puskureissa mahdollisesti ollut tieto jää kirjoittamatta levylle, jolloin kesken olleiden tiedostojen pituudeksi tulee nolla ja ne näyttävät tyhjiltä tai sitten niistä syntyy orpoja varausyksiköitä. Pahimmassa tapauksessa jopa levyn kirjanpito saattaa vahingoittua.

Välimuistiohjelma tulee helposti ladattua muistiin liian aikaisin, sillä esimerkiksi Windows 3.1:n asennusohjelma lisää sen aina AUTOEXEC.BATin ensimmäiseksi riviksi. Myös käyttäjä sijoittaa mielellään Smartdrv:n latauksen heti AUTOEXEC.BATin alkuun, koska näin loppuosa komentojonosta toimii nopeammin. AUTOEXEC.BATin alun pitäisi kuitenkin näyttää seuraavalta:

```
PATH C:\DOS;C:\WINDOWS
KEYB SU,,C:\DOS\KEYBOARD.SYS
SMARTDRV 1024 512 C+
(...tähän loput AUTOEXEC.BATin komennot)
```

DOSin tai Smartdrv:n ohjeet eivät kerro näin tärkeästä asiasta mitään ilmeisesti siksi, etteivät ohjelman tehneet amerikkalaiset tarvitse KEYB-ohjainta eivätkä he siksi ole tulleet edes ajatelleeksi että KEYBin ja Smartdrv:n keskinäisellä latausjärjestyksellä voisi olla jotain merkitystä.

44 B

Toinen ansa liittyy Smartdrv:n kirjoituspuskurin käyttöön levykkeillä. Koska levykkeen kirjoitussuojausta voi muuttaa käytön aikana ja levykkeen voi jopa poistaa asemasta kesken kaiken, kirjoituspuskuria ei pitäisi koskaan käyttää. Oletusarvona onkin, että Smartdrv nopeuttaa ainoastaan levykkeiden lukemista, ei niille kirjoittamista.

Jos halutaan käyttää myös kirjoituspuskuria, annetaan komento

```
SMARTDRV A+
```

joka ottaa käyttöön A: aseman kirjoituspuskurin. Tämä voi kuitenkin tuottaa vakavia ongelmia ainakin Smartdrv 4.0-versiossa.

Esimerkki: kun A: asemaan laitetaan kirjoitussuojattu levyke ja sille yritetään kopioida tiedostoja komennolla COPY *.* A:, kirjoitus käynnistyy normaalisti sillä COPYn kirjoitukset menevät ensin Smartdrv:n puskuriin. Vasta kun viisi sekuntia on kulunut kirjoituksen loppumisesta tai kun Smartdrv:n kirjoituspuskuri on tullut täyteen, se yrittää kirjoittaa puskuriin tulleet tiedot levykkeelle. Suojauksen vuoksi kirjoitus ei tietenkään onnistu, joten Smartdrv esittää ruudun ylänurkassa varoituksen. Se pyytää poistamaan kirjoitussuojan ja painamaan R:ää, jotta kirjoitusta voidaan yrittää uudelleen.

Tässä tilanteessa Smartdrv hyväksyy vain ja ainoastaan R-näppäimen. Koska levylle tarkoitettu aineisto on yhä ohjelman puskurissa, se ei anna esimerkiksi keskeyttää kirjoitusta Abortilla, kuten DOS tekisi vastaavassa tilanteessa. Jopa Ctrl+Alt+Del on tilapäisesti estetty.

Ainoaksi vaihtoehdoksi jää laittaa asemaan suojaamaton levyke, jolle Smartdrv saa purkaa tietonsa — tai katkaista koneesta sähköt, mikä johtaa tietojen menetykseen.

Komentotulkki ja apuohjelmat

45

Ohjelmatiedostojen suoritusjärjestys

Jotta komentotulkki käynnistäisi tiedoston, sen tarkentimena on oltava joko COM, EXE tai BAT. Tällöin riittää, että kirjoitetaan pelkkä ohjelman nimi. Komentotulkki lisää tarkentimen automaattisesti.

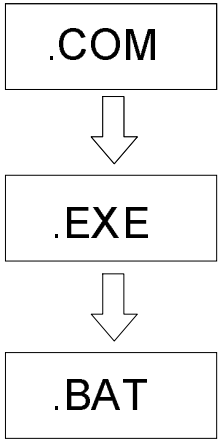
Mutta entäs sitten, jos samassa hakemistossa on useita saman nimisiä, mutta tarkentimiltaan poikkeavia tiedostoja? Jos hakemistossa ovat sekä tiedostot WIN.COM, WIN.EXE että WIN.BAT niin mikä niistä käynnistyy? Vai onko tarkentimella lainkaan merkitystä eli käynnistääkö komentotulkki sen, joka sattuu olemaan DIR-listauksessa ensimmäisenä?

Kyllä tarkentimella on merkitystä. Hakemistoa tutkiessaan DOS etsii ensin COM-päätteistä ohjelmaa. Ellei sitä löydy, etsitään EXE-tarkenninta. Ellei sitäkään löydy, kokeillaan vielä BATia. Komentojono on siis vasta hierarkian viimeisenä ja käynnistyy vain, jos hakemistossa ei ole saman nimistä COM- tai EXE-tiedostoa.

Kun yksi hakemisto on käsitelty, etsintä jatkuu muista PATHin osoittamista hakemistoista siinä järjestyksessä kuin ne on lueteltu PATHin rivillä. Vasta sitten, kun kaikki hakemistot on käyty läpi COM-EXE-BAT järjestyksessä eikä tiedostoa ole silti löytynyt, tulostuu tuttu ilmoitus

```
Bad command or file name
```

Suoritusjärjestyksellä on joskus merkitystä. Jos esimerkiksi erehtyy kopioimaan samaan hakemistoon sekä FORMAT.COMin että FORMAT.EXEn, komento FORMAT käynnistää aina COM-version.



Kun komentotulkki etsii käynnistettävää ohjelmaa, se yrittää ensin löytää COM-tiedoston. Seuraavaksi etsitään EXE-tyyppistä tiedostoa ja vasta viimeisenä vaihtoehtona BATia.

Näin voi helposti käydä kun päivittää DOS-versioita, sillä eri DOS-versioissa COM- ja EXE-tarkentimet saattavat vaihdella. Varminta onkin tyhjentää koko DOS-hakemisto vanhoista apuohjelmista ennen uusien kopioimista.

Eräät viruksetkin käyttävät tätä hyödykseen. Ne "tarttuvat" vain EXE-ohjelmiin luomalla levylle saman nimisen COM-tiedoston ja usein vielä piilottamalla sen. Kun käyttäjä sitten kirjoittaa ohjelman nimen, COM-tiedoston virus käynnistyy ensin ja vasta kun virus on tehnyt mitä haluaa, se käynnistää EXE-tiedoston ja luovuttaa ohjauksen sille. Jollei käyttäjä ole tarkkana, hän ei huomaa koko asiaa.

Toinen tilanne, missä suoritusjärjestyksellä on merkitystä, liittyy komentojonoihin. Jos esimerkiksi halutaan korvata Windowsin käynnistyskomento saman nimisellä komentojonolla ja sijoittaa komentojono samaan hakemistoon WIN.COMin kanssa, on jomman kumman tiedoston nimeä pakko vaihtaa. Muuten kävisi niin, että hakemistosta käynnistyisi aina vain WIN.COM eikä komentojonoa saisi mitenkään käyntiin, ei edes kirjoittamalla WIN.BAT.

46

Tiedostojen ja hakemistojen piilottaminen

Tiedoston piilottaminen tehdään kytkemällä sen piilomääre voimaan. DOS 5.0:sta alkaen työn voi tehdä ATTRIB-ohjelmalla. Komento

```
ATTRIB +H TEKSTI.TXT
```

poistaa tiedoston näkyvistä ja komento

```
ATTRIB -H TEKSTI.TXT
```

muuttaa sen taas näkyväksi. Pelkkä komento

```
ATTRIB *.*
```

näyttää kaikki hakemistossa olevat tiedostot, myös piilotetut. Saman tekee DIR /AH. Jos DOS-versio on vanhempi kuin 5.0, piilomääreen kytkeminen pitää tehdä joko PC Toolsin, Nortonin tai jonkin muun apuohjelman avulla.

Kokonainen hakemisto piilotetaan samalla tavalla:

```
ATTRIB +H SALAINEN
```

Kun hakemisto on piilotettu, se ei näy DIR-listauksessa eikä sitä näytä edes ATTRIB *.* joka käsittelee vain tiedostoja. Ainoa komento, joka näyttää myös piilotetut hakemistot, on DIR /AH.

Piilotuksesta huolimatta hakemistoa voidaan käyttää normaaliin tapaan. Esimerkiksi komento

```
CD SALAINEN
```

toimii, vaikka hakemisto olisi piilotettukin. Tällaista menettelyä käyttävät monet UNDELETE-ohjelmat, jotka siirtävät "poistetut" tiedostot piilotettuun hakemistoon ja tyhjentävät hakemiston vasta erillisellä käskyllä.

DOS 5.0:aan saakka CHKDSK pitää kuitenkin hakemiston piilotusmäärettä virheenä ja varoittaa asiasta:

```
C:\TEKSTIT\SALAINEN
   Invalid sub-directory entry
```

Ellei tiedoston näkyminen haittaa, sen ajo tai muu käyttö voidaan estää seuraavan kohdan nikseillä.

47

Ohjelmien käytön estäminen

Yksinkertaisin tapa estää ohjelman käynnistäminen on määritellä saman niminen makro, joka tulostaa virheilmoituksen. Tämä edellyttää DOSKEYn käyttöä, joka kuuluu DOSin mukaan vasta DOS 5.0:sta alkaen. Komento

```
DOSKEY FORMAT=REM Et voi käyttää FORMAT-ohjelmaa!
```

määrittelee FORMAT-nimisen makron, joka tulostaa kuvaruudulle virheilmoituksen aina kun käyttäjä yrittää käynnistää ohjelman. Esto ei ole kovin tehokas, sillä vähänkin DOSia tunteva käyttäjä huomaa nopeasti mistä on kyse ja osaa antaa komennon

```
DOSKEY FORMAT=
```

jonka jälkeen makro on purettu ja FORMAT-komento toimii jälleen normaalisti. Itse asiassa edes makron purkamista ei tarvita, sillä pelkkä välilyönnin lisääminen FORMAT-komennon eteen riittää erottamaan sen saman nimisestä makrosta ja käynnistämään ohjelman.

47 B

Toinen, tehokkaampi keino on vaihtaa ohjelman nimi siten, että se täsmää jonkin komentotulkin sisäisen käskyn kanssa. Jos esimerkiksi ohjelman nimi muutetaan ensin kirjoittamalla

```
REN FORMAT.COM CD.COM
```

sen käynnistys ei enää onnistu, koska CD viittaa saman nimiseen komentoon. Edes yritys käynnistää ohjelma kirjoittamalla

```
C:\DOS>CD.COM
```

ei onnistu, koska komentotulkki tulkitsee rivin edelleenkin hakemiston vaihtokäskyksi.

Miten ohjelman sitten saa käyntiin?

Niksi piilee siinä, että kun käynnistyskomennon eteen lisätään polkumääre, DOS ymmärtää kyseessä olevan ohjelman eikä hakemiston ja komento

```
C:\> C:\DOS\CD
```

käynnistää ohjelman.

47 C

Viimeinen tapa estää ohjelman käynnistys on lisätä sen nimen perään näkymätön tyhjämerkki, jonka ASCII-koodi on 255. Se saadaan aikaan pitämällä Alt-näppäintä pohjassa ja naputtelemalla numerot 2 5 5 *numeeriselta* erikoisnäppäimistöltä. Merkki näyttää tyhjältä, mutta ei ole sama kuin välilyönti. Siksi sen voi lisätä vaikkapa tiedostonimen keskelle. Ellei käyttäjä tiedä tyhjämerkin niksiä, hän ei pysty mitenkään avaamaan tiedostoa.

Jotta ohjelmanimen perään voitaisiin lisätä tyhjämerkkejä, nimen pituuden pitää tietenkin olla alle kahdeksan merkkiä. Koska tyhjämerkit eivät mitenkään näy DIR-listauksessa, käyttäjä voi vain ihmetellä miksi hakemistossa näkyvä ohjelma ei suostukaan käynnistymään nimensä mukaisesti.

Samaa periaatetta voi luonnollisesti käyttää myös työtiedostojen suojaamiseen. Eräät sovellukset saattavat kuitenkin itse tarkistaa tiedostonimen loogisen oikeellisuuden ja kieltäytyä hyväksymästä nimiä, joissa on muita kuin normaaleja näkyviä merkkejä.

48

Sisäisten käskyjen estäminen

Jos halutaan estää komentotulkin sisäisten käskyjen, kuten DEL-poistokomennon käyttö, on jälleen turvauduttava DOSKEYn apuun. Ellei sitä voida tai haluta käyttää, on ainoa mahdollisuus peukaloida suoraan komentotulkkitiedostoa.

Ohjelmatiedoston sisällön muuttaminen suoraan heksakoodeja käsittelemällä (englanniksi patch) on tarkkaa työtä. Jos koodeissa tulee tehtyä virheitä, ohjelma lakkaa toimimasta tai alkaa käyttäytyä omituisesti. Erityisen vaarallista tämä on juuri komentotulkin kohdalla, koska sen virheet saattavat estää käyttöjärjestelmää latautumasta ja siten virheitä ei päästä edes korjaamaan!

Ennen kuin komentotulkin peukalointia aloitetaan, on syytä tehdä käynnistyskelpoinen DOS-levyke ja varmuuskopioida alkuperäinen COMMAND.COM sekä levykkeelle että toiselle nimelle kiintolevyn sisällä. Jos COMMAND.COM sotkeentuu pahasti, DOS ei enää käynnisty kiintolevyltä ja ainoa tapa saada kone käyntiin on turvautua DOS-levykkeeseen.

Komentotulkki avataan muokkausta varten komennolla

```
DEBUG COMMAND.COM
```

Haluttu komento etsitään tämän jälkeen DEBUGin Search-komennolla. Jotta komentotulkin pituutta ei tarvitsisi laskea, oikaistaan kirjoittamalla sen paikalle FFFF joka vastaa 64 kilotavua.

Komento

```
-s 0000 FFFF "DEL"
```

tuo näkyville kaksi osoitetta, missä DEL esiintyy:

```
183E:8DD5
183E:A9EF
```

Kaksoispistettä ennen olevista luvuista ei tarvitse välittää, koska ne riippuvat siitä kohdasta muistia, jota DEBUG sattuu käyttämään. Lukuparin jälkimmäiset osat ovat sitävastoin tärkeitä.

Komento

```
-d 8DD5
```

näyttää pätkän muistista jolloin havaitaan, että kohdassa on /? -valitsimella saatava aputeksti. Jos DEL-käsky halutaan piilottaa käyttäjältä, aputekstiä ei tietenkään kannata muuttaa. Toinen kohta saadaan näkyviin komennolla

```
-d A9EF
```

jolloin DEBUG näyttää listan COMMAND.COMin sisäisten komentojen nimistä. DELin paikalle voidaan kirjoittaa vaikkapa POI

```
-e A9EF "POI"
```

Uusien komentonimien on oltava yhtä pitkiä kuin vanhojenkin, jotta komentotulkin ohjelmakoodi ei menisi sekaisin. Komentonimet on kirjoitettava isoilla kirjaimilla, sillä DOS muuntaa näppäimistöltä annetut komennot isoiksi ennen kuin se alkaa tulkita niitä.

Samalla periaatteella tehdään tarvittavat muutkin vaihdokset. Sen jälkeen muokattu komentotulkki tallennetaan takaisin levylle komennolla

```
-w
```

ja DEBUG lopetetaan kirjoittamalla

```
-q
```

Kun kone käynnistetään uudelleen, yritys poistaa tiedostoja kirjoittamalla

```
DEL TEKSTI.TXT
```

tuottaa virheilmoituksen Bad command or file name ja vain komento

```
POI TEKSTI.TXT
```

toimii halutulla tavalla.

49

Kirjoitusvaivan vähentäminen

Kun DOS-komentoja käyttää toistuvasti päivästä toiseen, tulee helposti miettineeksi, miten kirjoitusvaivaa voisi vähentää. Onko mitään, jonka voisi jättää kirjoittamatta ilman, että komennon merkitys muuttuu?

Helpointa on korvata merkintä *.* yksinkertaisesti pisteellä. Piste on oletushakemiston lyhenne ja hakemiston kaikkiin tiedostoihin voidaan usein viitata pelkällä hakemiston nimellä. Esimerkiksi tuttu komento

```
COPY A:*.*
```

voidaan kirjoittaa lyhyemmin

```
COPY A:.
```

Oletushakemiston tyhjentäminen käy vastaavasti kirjoittamalla

```
DEL .
```

Kun viitataan muihin hakemistoihin, riittää usein pelkkä hakemiston nimi. On esimerkiksi sama kirjoitetaanko

```
COPY C:\ALI A:
```

vai ohjesäännön ja opetuksen mukaisesti

```
COPY C:\ALI\*.* A:
```

Ensimmäinen oikotie on neljä merkkiä lyhyempi ja varsinkin hankalan kenoviivan poisjääntі helpottaa kirjoittamista.

Toinen samalla tavalla toimivia apuohjelma on XCOPY.

Aina tämä ei kuitenkaan onnistu. Esimerkiksi ATTRIBin kohdalla yritys kirjoittaa

```
ATTRIB +R ALI
```

ei kirjoitussuojaa kaikkia ALI-hakemiston tiedostoja vaan asettaa kirjoitussuojauksen pelkälle hakemistonimelle. Eikä edes se toimi: RD-komento poistaa hakemiston mukisematta, vaikka kirjoitussuojaus olisikin päällä. XCOPY käyttäytyy vastaavassa tilanteessa vielä omituisemmin ja hyppää tyynesti kaikkien +R:llä suojattujen hakemistojen ylitse. DOSin tekijät eivät selvästikään ole tulleet ajatelleiksi, että joku yrittäisi suojata hakemistojen nimiä.

Vanhat (ennen DOS 5.0:aa tulleet) ATTRIB-versiot eivät hyväksy edes muotoa

```
ATTRIB .
```

korvaamaan *.* merkintää. DOS 5.0:sta alkaen ATTRIBia on parannettu ja kaikkien hakemiston tiedostojen suojaamiseksi riittää että kirjoitetaan

```
ATTRIB +R
```

siinä hakemistossa, missä suojaus halutaan. Vastaavasti kaikkien tiedostojen suojaus ja piilotus poistetaan komennolla

```
ATTRIB -R -H
```

Tämä on nopea tapa varmistaa, ettei hakemistossa ole piilotettuja tai suojattuja tiedostoja jotka estäisivät sen poistamisen.

XCOPYn tapauksessa riittää usein pelkkä aseman nimi eikä hakemistoviittausta tarvita lainkaan. Jos esimerkiksi halutaan kopioida kaikki A: aseman tiedostot oletushakemistoon, riittää kun kirjoitetaan

```
XCOPY A:
```

Aivan kuten COPYn tapauksessa, kaikki hakemiston tiedostot saadaan kopioitua kirjoittamalla

```
XCOPY C:\KUVAT A:
```

eikä hankalaa *.* -merkintää tarvita.

50

Uskaltaako Backup-ohjelmaa käyttää?

DOSin oma varmuuskopiointiohjelma, Backup, kärsii yhä huonosta maineesta jonka se sai ensimmäisissä DOS-versioissa. Jopa Microsoft itse dokumentoi aikanaan 22 erilaista bugia Backup/Restore-ohjelmaparin toiminnasta. Osa bugeista oli suorastaan huvittavia: esimerkiksi MS-DOS 3.2 Backup kirjoittaa varmistustiedostoihin kenoviivojen tilalta kauttaviivoja, jolloin PC-DOS 3.2 ei tietenkään tunnista niitä ja päinvastoin.

Versiosta 5.0 lähtien kaikkien bugien pitäisi olla korjattu, mutta ohjelmissa on silti omat heikkoutensa. Ne ovat hitaita, eivät pakkaa tiedostoja eikä Backup numeroi varmistussarjoja erikseen. Kun Restore pyytää levykettä numero kolme, se hyväksyy minkä tahansa kolmoslevykkeen — jopa aivan eri koneessa ja eri varmistusajossa tehdyn.

Vasta 5.0-versiosta alkaen Restoren pitäisi pystyä palauttamaan myös vanhemmilla DOS-versioilla tehtyjä varmistusajoja. Tätä vanhemmat ohjelmat ovat versiosidonnaisia eivätkä välttämättä pysty palauttamaan edes vanhemmilla DOSeilla tehtyjä varmistuksia, uudemmista puhumattakaan. Backupilla tehty varmistus voidaan luotettavasti palauttaa vain samaa DOS-versiota olevalla Restorella. *Tästä syystä Backup/Restorea ei koskaan pidä käyttää DOS-päivityksen yhteydessä!*

Kaikesta huolimatta Backupilla on omat hyvätkin puolensa. Jos esimerkiksi tehdään varmistuksia isolle irrotettavalle levylle (kuten optiset levyt, Bernoulli tai 20 megatavun floptical-korput), tiedostojen kopiointi kiintolevyltä levylle tapahtuu huomattavasti nopeammin Backupilla kuin COPYllä tai edes XCOPYllä ja vie myös hieman vähemmän tilaa. Tämä johtuu siitä, että Backup kirjoittaa kaikki tiedostot peräkkäin yhdeksi isoksi tiedostoksi, jolloin kirjanpidon käytöstä aiheutuva viive jää lähes kokonaan pois ja varausyksiköiden hukkatila minimoituu. Varjopuolena on, ettei yhdelle levylle mahdu kuin yksi varmistusajo, sillä Backup luo varmistustiedoston aina samalle nimelle ja päähakemistoon. Itse asiassa se tyhjentää koko päähakemiston.

Versiosta 6.0 alkaen vanhan merkkipohjaisen Backup/Restoren käyttöön on tuskin mitään syytä, koska DOSin mukaan laitettu varmuuskopiointiohjelma on paljon tehokkaampi, näppärämpi ja helpompi käyttää.

51

Kuvaruudun värimääritykset

ANSI-ohjauksen avulla kuvaruudun taustalle ja tekstille saadaan määriteltyä halutut värit. Ne tuovat piristystä käyttäjälle silloin, kun tavanomainen valkoinen teksti mustalla pohjalla alkaa kyllästyttää.

Kaikki ANSI-määritykset tehdään samalla periaatteella: kuvaruudulle pitää saada tulostettua teksti, jossa on ensin Esc-merkki (koodinumero 27), hakasulku ja sen jälkeen erilaisia numerokoodeja. Tekstin tulostaminen onnistuu joko kirjoittamalla määritykset tiedostoon ja tulostamalla se ruudulle TYPE-käskyllä tai sitten käyttämällä PROMPT-komentoa.

Esc-koodin kirjoittaminen tiedostoon vaatii pientä kikkailua, kuten kohdassa 53 on esitetty. Tarvittavien koodien kirjoittaminen PROMPTin avulla on helpompaa, koska hankalan Esc-merkin voi lyhentää koodilla $e.

Värimääritys kirjoitetaan muodossa

```
Esc[t,vm
```

missä t on taustavärin koodinumero ja v merkkien värin koodinumero oheisen taulukon mukaisesti. Värimääritys päätetään aina m-kirjaimeen. Esimerkiksi

```
Esc[47;34m
```

ottaa käyttöön valkoisen taustan (47) ja sen päälle sinisen tekstin (34). ANSI-ohjaimen tuntemat värikoodit on esitetty seuraavassa taulukossa:

Väri	Taustakoodi	Merkkivärin koodi
Musta	40	30
Punainen	41	31
Vihreä	42	32
Keltainen	43	33
Sininen	44	34
Violetti	45	35
Vaaleansininen	46	36
Valkoinen	47	37

Jos määritys on kirjoitettu tiedostoon, se saadaan käyttöön tulostamalla tiedosto ruudulle TYPE-komennolla. PROMPTin avulla sama määritys kirjoitetaan

```
PROMPT $e[47;34m
```

Rivin perään pitäisi vielä lisätä tavanomainen valmiusmerkki (kuten pg), jotta sekin tulisi huomioitua värimääritysten ohella. PROMPTissa olevat määritykset tulostuvat ruudulle aina kun valmiusmerkkikin.

Käytettiinpä sitten kumpaa tapaa tahansa, valitut värit täyttävät näytön joko seuraavien komentojen tekemän tulostuksen myötä tai kerralla, jos ruutu tyhjennetään CLS-komennolla. Niiden vaikutus ei kuitenkaan ulotu sovellusohjelmiin, jotka kirjoittavat suoraan kuvaruutumuistiin ja ohittavat ANSI-ohjaimen. Näissä sovelluksissa värejä voi muuttaa vain, mikäli ohjelmissa on oma värien määritystoimintonsa.

Värien ohella ANSI:lla saadaan ohjattua myös tekstiattribuutteja oheisen taulukon mukaisesti. Esimerkiksi

```
PROMPT $e[1m$p$g$e[0m
```

saa valmiusmerkin näkymään muuta komentoriviä kirkkaampana. ANSI-koodien lisäksi PROMPT-määrityksestä ei saa unohtaa tavallistakaan valmiusmerkkiä (pg).

Vielä selkeämmin erottuu vilkkuva valmiusmerkki, joka syntyy komennolla

```
PROMPT $e[5m$p$g$e[0m
```

Attribuutti	Esc-komento
Attribuuttien poisto	Esc[0m
Kirkkaus	Esc[1m
Alleviivaus yksivärinäytöllä	Esc[4m
Vilkutus	Esc[5m
Käänteinen	Esc[7m
Musta	Esc[8m

52

Näppäinten uudelleenmääritys

ANSI-ohjausta voi käyttää värivalintojen lisäksi myös näppäinten toiminnan uudelleenmääritykseen. Ohjelmoimalla usein tarvittavat DOS-komennot vaikkapa funktionäppäimille voidaan kirjoitustyötä vähentää — mutta puuhassa on myös omat vaaransa.

Määritys tehdään edellisen niksin tapaan, mutta nyt muodossa

```
Esc[koodi;"Teksti"p
```

Koodin paikalle kirjoitetaan uudelleen määriteltävän näppäimen nimi. Teksti, joka siitä halutaan tulevan, kirjoitetaan lainausmerkkien sisään. Lopussa oleva p jättää määrityksen muistiin.

Esimerkiksi määritys

```
Esc[0;59;"DIR"p
```

sijoittaa tekstin DIR näppäimen F1 taakse. Enterin painallus saadaan lisättyä koodinumerolla 13, jolloin määritys näyttää seuraavalta:

```
Esc[0;59;"DIR";13p
```

Nyt pelkkä F1-näppäimen painallus tuo näyttöön oletushakemiston DIR-listauksen. Komento, joka näyttää DIR-listauksen A: asemasta Alt+A:ta painamalla, ohjelmoidaan seuraavasti:

```
Esc[0;32;"DIR A:";13p
```

Eri toimintonäppäinten koodit saadaan seuraavasta taulukosta:

Näppäin	Paljas	Shift+	Ctrl+	Alt+
F1	0;59	0;84	0;94	0;104
F2	0;60	0;85	0;95	0;105
F3	0;61	0;86	0;96	0;106
F4	0;62	0;87	0;97	0;107
F5	0;63	0;88	0;98	0;108
F6	0;64	0;89	0;99	0;109
F7	0;65	0;90	0;100	0;110
F8	0;66	0;91	0;101	0;111
F9	0;67	0;92	0;102	0;112
F10	0;68	0;93	0;103	0;113
F11	0;133	0;135	0;137	0;139
F12	0;134	0;136	0;138	0;140

Myös Alt+merkki näppäinyhdistelmät voidaan määritellä uudelleen oheisen taulukon koodeja käyttäen:

Alt+A	0;30	Alt+Y	0;21
Alt+B	0;48	Alt+Z	0;44
Alt+C	0;46	Alt+0	0;129
Alt+D	0;32	Alt+1	0;120
Alt+E	0;18	Alt+2	0;121
Alt+F	0;33	Alt+3	0;122
Alt+G	0;34	Alt+4	0;123
Alt+H	0;35	Alt+5	0;124
Alt+I	0;23	Alt+6	0;125
Alt+J	0;36	Alt+7	0;126
Alt+K	0;37	Alt+8	0;127
Alt+L	0;38	Alt+9	0;128
Alt+M	0;50	Alt+-	0;130
Alt+N	0;49	Alt+=	0;131
Alt+O	0;24	Nuoli ylös	0;72
Alt+P	0;25	Nuoli alas	0;80
Alt+Q	0;16	Nuoli vasemmalle	0;75
Alt+R	0;19	Nuoli oikealle	0;77
Alt+S	0;31	End	0;79
Alt+T	0;20	Home	0;71
Alt+U	0;22	Ins	0;82
Alt+V	0;47	Del	0;83
Alt+W	0;17	PgDn	0;81
Alt+X	0;45	PgUp	0;73

Näppäimistömääritysten vaara piilee siinä, että pilailu- tai vahingontekomielessä niitä voidaan upottaa tavallisiin tekstitiedostoihin, jolloin ne astuvat voimaan kun käyttäjä tulostaa tiedoston pahaa-aavistamatta kuvaruudulle. Tällaisia temppuja kutsutaan myös ANSI-troijalaisiksi. Niiltä välttyy varmimmin kun jättää ANSI.SYSin asentamatta.

53

Escape-koodin kirjoittaminen

Esc-koodeja tarvitaan paitsi ANSI-ohjauksissa myös lähetettäessä kirjoittimelle sen toimintaan vaikuttavia ohjauskoodeja. Esimerkiksi koodisarja Esc&l2X kytkee PCL-kielessä käyttöön toimintatilan, jossa jokainen sivu tulostuu kahtena kappaleena.

Ongelmana on Esc-koodin kirjoittaminen. Sitä ei voi tehdä suoraan näppäimistöltä, koska Esc-näppäimen painallus tulkitaan toiminnon keskeyttämiseksi. Edes Alt+027 ei auta, vaan sekin keskeyttää kirjoituksen.

Joissakin tekstinkäsittelyohjelmissa on mahdollista upottaa tekstiin Esc-merkki, mutta useimmissa ei. Onneksi DOSin EDIT pystyy tähän, vaikka vanhempi EDLIN ei. Esc-merkki kirjoitetaan painamalla ensin Ctrl+P, jonka jälkeen EDIT jää odottamaan seuraavaa näppäintä ja kun painetaan Esc-näppäintä, sen koodi menee suoraan tiedostoon. Merkkinä Escapesta ruudulla näkyy pieni taaksepäin osoittava nuoli. Pelkän Esc-näppäimen painalluksen EDIT tulkitsee keskeytyskomennoksi eikä hyväksy sitä itse tiedostoon.

Tällä niksillä voidaan luoda esimerkiksi tiedosto TUPLA.OHJ, joka sisältää PCL-kielen käyttämän koodisarjan. Kun se tulostetaan kirjoittimelle komennolla

```
COPY TUPLA.OHJ /B LPT1:
```

kirjoitin alkaa tulostaa sivut kahtena. Vastaavalla tavalla saadaan aikaan muutkin yleisesti tarvittavat ohjauskomennot. Jos niitä on paljon, jokainen komento kannattaa kirjoittaa omaksi komentojonoksi.

54

VER /R

Komentotulkin sisäinen VER-käsky näyttää käytössä olevan DOSin versiotiedot:

```
C:\DOS>VER
MS-DOS Version 5.00
```

Monissa DOS-versioissa käsky tuntee dokumentoimattoman /R-valitsimen, joka kertoo myös DOSin sisäisen revisionumeron sekä tiedon siitä, onko DOS siirretty HMA-alueelle vai onko se perusmuistissa:

```
C:\DOS>VER /R

MS-DOS Version 5.00
Revision A
DOS is in HMA
```

55

FORMATin piilotetut valitsimet

FORMAT-ohjelma tuntee dokumentoimattoman /BACKUP-valitsimen. Kun sitä käytetään, FORMAT ei esitä turhia "Are you sure"-kysymyksiä vaan alkaa alusta levyä heti. Alustuksen jälkeen FORMAT kysyy vain levyn nimen eikä ehdota uuden levykkeen alustusta. Valitsin on nimensä mukaisesti tarkoitettu BACKUP-ohjelmalle, joka käyttää sitä kutsuessaan FORMATia alustamaan levykkeitä varmuuskopioinnin aikana. /BACKUP toimii vasta DOS 4.0:sta alkaen, mutta DOS 3.3:ssa on vastaava ominaisuus piilotettuna /H -valitsimen taakse.

Tätä voi käyttää hyväksi esimerkiksi omassa FORMAT-ohjelman korvaavassa komentojonossa. Varsinainen alustusohjelma FORMAT.

COM nimetään aluksi uudelleen vaikkapa FMAT.COMiksi. Sen jälkeen kirjoitetaan komentojono FORMAT.BAT:

```
ECHO OFF
IF %1==a: GOTO OK
IF %1==A: GOTO OK
ECHO Levykkeitä voi alustaa vain A: asemassa!
GOTO LOPPU
:OK
FMAT A: /BACKUP %2 %3
:LOPPU
```

FORMATissa on muitakin piilotettuja valitsimia.

/AUTOTEST poikkeaa /BACKUPista vain siinä, ettei se kysy edes levykkeen nimeä. Alustaminen tapahtuu täysin ilman käyttäjän apua.

/SELECT ei DOS 5.0:sta alkaen tee enää mitään. Se varmistaa kirjanpidon ajamalla MIRRORin, mutta ei kuitenkaan tee alustusta. Valitsin on peräisin vanhoista DOSeista, joissa ollut SELECT-komento loi maakohtaiset CONFIG.SYS-asetukset sisältävän käynnistyslevykkeen.

55 B

Myös muissa apuohjelmissa on piilotettuja valitsimia. DOS 4.0:ssa ATTRIB tuntee mm. parametrit DATE, TIME ja FILESIZE, joilla se näyttää vastaavat tiedot käsittelemistään tiedostoista. Nämä parametrit on kuitenkin poistettu DOS 5.0:sta alkaen eikä niitä enää tarvittaisikaan, koska DIR-käsky tekee saman asian.

56

Useita komentoja yhdellä rivillä

DOSin oma komentotulkki hyväksyy vain yhden komennon kerrallaan. Tätä rajoitusta on kierretty 4DOSissa ja OS/2:ssa, mikä onkin hyvä, sillä joskus olisi mukava kirjoittaa saman tien kaikki tarvittavat komennot. Varsinkin silloin kun tiedetään, että komentojen suorittaminen tulee

kestämään oman aikansa, olisi hyvä jos ne voisi kirjoittaa jo etukäteen valmiiksi.

Tämäkin on mahdollista, mutta edellyttää pientä kikkailua. Niksin ydin on putkikomennon käyttö. Yleensähän putkea käytetään silloin, kun ensimmäisen ohjelman lähettämä syöte halutaan antaa toiselle ohjelmalle, sen syöte edelleen kolmannelle jne. Putkimerkillä voidaan erottaa komennot silloinkin, kun ne eivät joko lähetä mitään tulostusta eteenpäin tai kun tulostuksella ei ole mitään merkitystä seuraavan komennon kannalta.

Jos esimerkiksi halutaan perustaa neljä hakemistoa yhdellä rivillä, voidaan kirjoittaa

```
C:\>MD eka | MD toka | MD kolmas | MD neljäs
```

Jos taas halutaan käynnistää aikaa vievä kopiointisarja verkossa, kirjoitetaan

```
C:\TEKSTI>COPY *.TXT Z:|COPY *.DOC Y:|COPY *.WP5 W:
```

Kaikkien käynnistyskelpoisten tiedostojen tuhoaminen puolestaan tapahtuu seuraavasti:

```
C:\TEMP>DEL *.EXE | DEL *.COM | DEL *.BAT
```

Niksi on hyödyllinen, mutta sen käytössä on omat rajoituksensa. Sitä ei voi käyttää silloin, kun komento pysähtyy kysymään jotain käyttäjältä. Esimerkiksi FORMAT A: tai DEL *.* -käskyjä ei voi käyttää, koska ne edellyttäisivät näppäimistöltä annettavaa kuittausta. Myös sellaiset komennot, joiden tekemä tulostus haluttaisiin nähdä (kuten DIR) voi unohtaa, koska niiden tulostus menee putkeen eikä ruudulle. Poikkeuksen muodostaa ainoastaan rivin viimeinen komento, joka ohjautuu normaaliin tapaan näytölle koska se sijaitsee putken päässä. Siten esimerkiksi komento

```
C:\>MD ALI | COPY \TEMP ALI | DIR ALI\*.DOC
```

toimii kuten käyttäjä haluaakin.

57

Virheilmoitus "Packed file is corrupt"

Jos ohjelman käynnistys tuottaa virheilmoituksen "Packed file is corrupt", ohjelmassa ei ole mitään vikaa. Syynä on ohjelman tekemiseen käytetty Microsoftin LINK-ohjelma, jonka pakkausvalitsin on aikanaan toiminut väärin. Se on tiivistänyt ohjelman koodia ja lisännyt siihen purkuosan, joka ei kuitenkaan toimi 64 kiloa pienemmässä muisti-osoitteessa.

Vika ei ole tullut ajoissa ilmi, koska DOS on vienyt hyvinkin yli 64 kiloa ja ohjelma on sijoittunut aina tämän maagisen rajan yläpuolelle. Vasta DOS 5.0 pystyi siirtämään osan koodistaan HMA-alueella, jolloin perusmuistista kului vain muutamia kymmeniä kiloja ja purkuohjelma sijoittui 64 kilon rajan alapuolelle.

Ongelman poistamiseksi DOS 5.0:sta lähtien alettiin toimittaa LOADFIX-nimistä apuohjelmaa, joka korjaa ohjelmatiedostossa olevan purkurutiinin. Kertakäsittely riittää — kun LOADFIX on ajettu, virheilmoitusta ei enää tule. Jos virheilmoituksen antava ohjelma on nimeltään SOFTA.EXE, annetaan komento

```
LOADFIX SOFTA.EXE
```

ja ongelma on poissa päiväjärjestyksestä.

Kunpa kaikki muutkin DOS-ongelmat olisivat yhtä helppoja ratkaista!

58

EDLIN ja DEBUG kauko-ohjattuina

EDLIN ja DEBUG pystyvät lukemaan syötteenä annetut komennot tekstimuotoisesta tiedostosta, johon ne on kirjoitettu etukäteen valmiiksi. Tämä mahdollistaa ohjelmien käytön "kauko-ohjattuina" ilman, että käyttäjän tarvitsee koskea näppäimistöön lainkaan.

Esimerkiksi komennot, jotka tarvitaan vaihtamaan tiedostossa esiintyvät echo-sanat vastaaviin isoihin kirjaimiin, kirjoitetaan VAIHTO.KOM-tiedostoon seuraavasti:

```
1,#Recho^ZECHO
e
```

Varsinainen tekstien vaihto tehdään sen jälkeen kirjoittamalla

```
EDLIN JONO.BAT <vaihto.kom
```

Vaihdon voi ulottaa kaikkiin .BAT-tiedostoihin käyttämällä toistokäskyä:

```
FOR %i IN (*.BAT) DO EDLIN %i <vaihto.kom
```

Johtuen kuitenkin tavasta, jolla DOS FOR-lauseen käsittelee, jokainen .BAT-tiedosto käsitellään itse asiassa kaksi kertaa koska tiedostoon tehdyn muutoksen jälkeen FOR-lause pitää sitä kokonaan uutena tiedostona.

Vastaavalla tavalla voidaan automatisoida mitä tahansa tekstitiedostoihin tehtäviä muutoksia. Esimerkiksi verkkokäytössä tukihenkilö voi yön aikana lisätä muutoskäskyn työasemien AUTOEXEC.BATiin, jolloin jokainen työasema tekee käynnistyessään itse muihin komentojonoihin tarvittavat muutokset.

Toinen tapa käyttää tekniikkaa on esitetty niksissä 121. Siinä muokataan komentojonoa EDLINin avulla ennaltaohjelmoidusti niin, että kaikkien rivien alkuun lisätään tiedoston poistokäsky.

Mahdollisuus muokata komentojonoja EDLINin avulla kauko-ohjatusti avaa myös aukon lähiverkon suojauksiin. Komentojonoon, joka sijaitsee esimerkiksi kaikille yhteisessä hakemistossa voidaan tällä periaatteella lisätä rivit, jotka muokkaavat jotakin käyttäjän omissa, suojatuissa hakemistoissa olevia komentojonoja arvaamattomilla ja jopa vahingollisilla tavoilla. Suojaukset ohittuvat, koska käyttöjärjestelmästä katsoen näyttää siltä kuin käyttäjä tekisi itse muutokset.

Kannattaa siis olla varovainen myös komentojonoja ajaessaan!

58 B

Vielä suurempaa hyötyä valmiiksi kirjoitetusta syötteestä on DEBUG-ohjelman kanssa, jossa se tarjoaa tavan pienten konekielisten ohjelmien luomiseen. Tätä tekniikkaa on käytetty myöhemmin tässä kirjassa.

Tarvittavat käskyt kirjoitetaan ensin tiedostoon, jonka tarkentimena on SRC (source eli lähdekielinen listaus). Käskyjen sarja voi näyttää esimerkiksi seuraavalta:

```
N BOOT.COM
A 0100
MOV AX,40
MOV DS,AX
MOV WORD PTR [72],1234
JMP FFFF:0000

RCX
10
W

Q
```

Tässä yhteydessä ei välitetä siitä, mitä kyseiset komennot tarkoittavat. Kuten myöhemmin selviää, kyseessä on ohjelma joka käynnistää koneen ohjelmallisesti uudelleen.

Ohjelma BOOT.COM syntyy, kun tehty SRC-tiedosto eli "skripti" ajetaan antamalla ne DEBUGille:

```
C:\KJONOT> DEBUG <tee-boot.src
-N BOOT.COM
-A 0100
1678:0100 MOV AX,40
1678:0103 MOV DS,AX
1678:0105 MOV WORD PTR [72],1234
1678:010B JMP FFFF:0000
1678:0110
-RCX
CX 0000
:10
-W
Writing 00010 bytes
-
-Q
```

Koska konekielisten muistikkaiden kirjoittaminen vaatii enemmän työtä kuin pelkkien numeroiden kirjoittaminen, edellä esitetyn tavan sijasta käytetään yleensä pelkkä heksakoodeja jotka syötetään muistiin skriptissä olevalla E-komennolla (Enter). Tällöin sama listaus näyttää hieman yksinkertaisemmalta:

```
N BOOT.COM
E 0100 B8 40 00 8E D8 C7 06 72
E 0108 00 34 12 EA 00 00 FF FF
RCX
10
W

Q
```

Tästä esitysmuodosta on myös se etu, että sitä voivat käyttää nekin joilla ei ole DOSin Debuggeria. Heksakoodit voi kirjoittaa suoraan .COM-tiedostoksi levylle esimerkiksi Nortonin DiskEditin (vanha NU) tai PC Toolsin avulla.

Debugin hyvin hallitseva käyttäjä voi kirjoittaa SRC-tiedostossa olevat heksakoodit suoraankin, jolloin aputiedostoa ei tarvita. Koska pienikin virhe riittää estämään ohjelman toiminnan ja todennäköisesti jumiuttaa koko mikron, kannattaa yleensä käyttää SRC-tiedoston apua.

59

Näppäimistön merkkivalot

PC:n näppäimistössä on kolme merkkivaloa: NumLock, CapsLock ja ScrollLock. Eräät DOS-versiot sytyttävät NumLock-valon automaattisesti kun mikro käynnistetään. Lisäksi monissa koneissa voidaan SETUPin avulla määritellä, onko valo päällä vai pois heti käynnistyksen jälkeen.

Valot voi sytyttää ja sammuttaa myös ohjelmallisesti. Ohjelma, joka luo CapsLock-valon hallintan tarvittavan ohjelman CAPSLOCK.COM, on esitetty seuraavalla sivulla.

```
N CAPSLOCK.COM
E 0100 B8 40 00 8E C0 B3 40 80
E 0108 3E 82 00 2D 74 06 26 08
E 0110 1E 17 00 C3 F6 D3 26 20
E 0118 1E 17 00 C3
RCX
001C
W
Q
```

Ohjelman käyttö on yksinkertaista. Kun valo halutaan päälle (ja isojen kirjainten lukitus voimaan), annetaan komento

```
CAPSLOCK +
```

ja kun halutaan palata perustilaan annetaan komento

```
CAPSLOCK -
```

Jos merkkivalo syttyy automaattisesti kun mikro käynnistyy, tämä komento sijoitettuna AUTOEXEC.BATin loppuun sammuttaa sen.

Vastaavalla tavalla voidaan käsitellä myös kahta muuta merkkivaloa. Niiden hallintaan tarkoitettu ohjelma on muuten täysin sama, mutta käsiteltävä bitti on eri kuin CapsLockin tapauksessa. Bitin vaatima muutos on näkyy ohjelmalistauksessa numerona 20 aiemman 40 tilalla.

NumLockin hallinta onnistuu seuraavalla ohjelmalla:

```
N NUMLOCK.COM
E 0100 B8 40 00 8E C0 B3 20 80
E 0108 3E 82 00 2D 74 06 26 08
E 0110 1E 17 00 C3 F6 D3 26 20
E 0118 1E 17 00 C3
RCX
001C
W
Q
```

Numerolukituksen voi kytkeä päälle vaikkapa samassa komentojonossa, joka käynnistää taulukkolaskennan.

```
ECHO OFF
ECHO Käynnistän Lotuksen, odota....
```

```
NUMLOCK +
LOTUS
NUMLOCK -
```

Viimeisenä esimerkkinä on vielä Scroll Lock-valoon liittyvä vastaava ohjelma, jossa bitistä kertova arvo on 10:

```
N SCRLOCK.COM
E 0100 B8 40 00 8E C0 B3 10 80
E 0108 3E 82 00 2D 74 06 26 08
E 0110 1E 17 00 C3 F6 D3 26 20
E 0118 1E 17 00 C3
RCX
001C
W
Q
```

Levyt

60

Korpulta lerpulle — ja takaisin

Sen jälkeen kun IBM siirtyi vuonna 1987 omissa mikroissaan yksistään korppuasemien käyttöön ovat markkinat jakautuneet sekä lerppu- että korppukoneisiin. Vaikka uudemmissa laitteissa onkin poikkeuksetta korppuasema, käytössä on yhä paljon lerppukoneita. Silloin ongelmaksi nousee tiedostojen siirtäminen korpulta lerpulle — ja takaisin.

Jotta siirto olisi mahdollista, yritykseen hankitaan vähintään yksi kone, jossa on sekä lerppu- että korppuasema. Sen jälkeen kuvitellaan, että tiedostojen siirto levykkeeltä toiselle käy käden käänteessä.

Valitettavasti mikään asia ei ole niin helppoa. Ei edes ATK-alalla.

Ensimmäisenä levykkeiden kopioija törmää siihen, ettei levykkeitä voikaan kopioida tavallisella DISKCOPY-ohjelmalla. Komento

```
DISKCOPY A: B:
```

näyttää kyllä aluksi toimivan, mutta kun tulee levykkeelle kirjoittamisen aika, kopiointi keskeytyy virheilmoitukseen. Korppua ei voi siirtää suoraan lerpulle koska DISKCOPY tekee levykkeestä täsmällisen jäljennöksen, eikä korppua voi jäljentää lerpulle niiden erilaisen koon ja erilaisen kirjanpidon vuoksi.

Seuraavaksi voi kokeilla tiedostojen siirtämistä perinteisellä COPY-käskyllä:

```
A:\>COPY *.* B:
```

Yleensä tämä komento riittää, mutta ei aina. Ensimmäinen ongelma voi tulla siitä, että jos korppu on 1,44-megainen ja lähes täynnä, sen tiedostot eivät yksinkertaisesti mahdu yhdelle 1,2-megaiselle lerpulle vaan

tiedostot pitää jakaa kahdelle eri levykkeelle. Ongelma on vielä todennäköisempi jos alkuperäinen korppu on 2,88-megainen.

Vaikka tilaongelmaa ei tulisikaan, COPY-käsky ei kopioi levykkeellä mahdollisesti olevia alihakemistoja. Yleensä levykkeillä ei käytetä alihakemistoja, koska ne hidastavat entisestään muutenkin hidasta levykkeen käsittelyä. Alihakemistoja saatetaan silti käyttää esimerkiksi tiedostojen järjestämiseen aihepiirin mukaan ja kuten myöhemmin ilmenee, alihakemistojen käyttö on ainoa tapa kiertää päähakemiston kokorajoitusta, joka muuten voi katkaista tiedostojen kopioinnin ennenaikaisesti.

Alihakemistoja voi siis olla, mutta onneksi niidenkin kopiointi on helppoa ja tapahtuu kirjoittamalla

```
A:\>XCOPY *.* B: /S
```

XCOPY ei kuitenkaan kopioi levykkeellä mahdollisesti olevia piilo- tai järjestelmätiedostoja. Tavallisilla levykkeillä niitä ei pitäisi olla, mutta jos kyseessä on suoraan ajettava ohjelmalevyke tai sillä on käyttöjärjestelmä, piilotetutkin tiedostot pitäisi saada siirrettyä.

Viimeistään kopioinnin jälkeen kannattaakin varmistaa, ettei levykkeelle ole jäänyt huomaamattomia, piilotettuja tiedostoja. DOS 5.0:sta lähtien tarkistuksen voi tehdä helpoiten kirjoittamalla

```
A:\>DIR /AH
```

mutta vanhemmilla DOS-versioilla joudutaan ajamaan CHKDSK ja tarkistamaan sen listauksesta, montako piilotettua tiedostoa levykkeellä on. Jos piilotettuja tiedostoja on vain yksi ja sen pituus on nolla, kyseessä on levykkeelle annettu nimi eikä varsinainen tiedosto:

```
0 bytes in 1 hidden file(s)
```

Mikäli piilotettuja tai järjestelmätiedostoja löytyy, ne pitää muuttaa ensin näkyviksi ja kopioida sen jälkeen. Uudella levykkeellä tiedostot pitää muistaa vielä palauttaa alkuperäiseen tilaansa. DOS 5.0:sta alkaen piilo- ja järjestelmämääreiden muuttaminen tapahtuu helposti ATTRIBilla, mutta vanhemmissa DOS-versioissa ei ole tähän sopivia työkaluja. DOS 4.0:sta lähtien tiedostomääreitä voidaan tosin muokata DOS Shellin kautta.

Jos kyseessä on käyttöjärjestelmälevyke tämäkään ei vielä riitä, sillä piilotetut käyttöjärjestelmätiedostot pitää sijoittaa oikeaan kohtaan levyä ja usein myös levykkeen käynnistyslohko pitäisi kirjoittaa uudelleen. Sitä ei voi edes kopioida suoraan korpulta lerpulle, koska käynnistyslohkon sisältämät tekniset parametrit ovat erilaisia korpuilla ja lerpuilla. Ainoa ohjelma, joka osaa siirtää käyttöjärjestelmätiedostot oikeaan paikkaan ja päivittää samalla käynnistyslohkon, on SYS.

60 B

Jos samasta levykkeestä pitää tehdä useita kopioita, kannattaa sen tiedostot kopioida ensin kiintolevylle alihakemistoon. Näin lähdelevyke, jonka käsittely on paljon hitaampaa kuin kiintolevyn käsittely, joudutaan lukemaan vain kerran. Vielä nopeammin kopiot syntyvät jos tiedostot kopioidaan aluksi RAM-levylle.

61

Miksei kopiolevyke kelpaa?

Vaikka kaikki korpun tiedostot olisi saatu tavalla tai toisella kopioitua lerpulle, se ei välttämättä toimi. Jos kyseessä on esimerkiksi jonkin ohjelman asennuslevykkeet, asennusta suorittava ohjelma ei tunnista levykettä oikein. Se pyytää asettamaan levykkeen numero kaksi asemaan, mutta vaikka käyttäjä näin tekee, asennusohjelma uudistaa itsepintaisesti pyyntönsä. Jokin on mennyt vikaan. Levykkeet eivät olekaan samanlaisia.

Ero on todennäköisesti levykkeiden nimissä. Asennusohjelma tunnistaa levykkeen sillä olevan nimen (volume label) perusteella ja kun nimi ei täsmää, se pitää levykettä vääränä. Nimi ei ole kopioitunut, koska DISKCOPY on ainoa ohjelma joka tekee levykkeestä niin tarkan kopion että myös nimi kopioituu.

Jotta asennus onnistuisi, kopiolevykkeiden nimet pitää käydä vaihtamassa samoiksi kuin mitä ne ovat alkuperäisillä levykkeillä. Vaihto

tapahtuu LABEL-komennolla. Valitettavasti asennuslevykkeiden nimet ovat joskus pelkkiä numero- ja kirjainyhdistelmiä, jolloin niiden kirjoittaminen on työlästä ja virhealtista.

Erot levykkeiden nimissä tulevat esille yleensä vain kun levykkeitä käytetään ohjelmien asentamiseen. Tavallisessa työtiedostojen tai ohjelmien siirtämisessä levykkeiden nimillä tai sarjanumeroilla ei ole merkitystä.

62

Miksi levykkeillä on sarjanumero?

DOS 4.0:sta alkaen FORMAT-ohjelma kirjoittaa jokaiselle alustamalleen levylle yksilöllisen sarjanumeron. Tämä numero näkyy DIR-listauksen alussa:

```
Volume in drive B has no label
Volume Serial Number is 227A-1401
Directory of B:\
```

Sarjanumero koostuu kahdesta heksadesimaalisesta luvusta, joten sarjanumerossa voi esiintyä numeroiden lisäksi myös kirjaimet A:sta F:ään.

Jos levyke kopioidaan DISKCOPYllä, ohjelma osaa jättää sarjanumeron kopioimatta ja keksii sen sijaan kopiolevykkeelle oman sarjanumeron. Jos kopiointi tehdään DOS 3.3:lla tai sitä vanhemmalla DOSilla, sarjanumerokin kopioituu koska ohjelmat eivät osaa jättää sitä pois. Tästä ei yleensä ole mitään vaaraa, sillä todennäköisyys sille että levykkeet sekoaisivat käytössä on hyvin pieni.

Sarjanumeroa tarvitaan, koska sen avulla DOSin on helpompi erottaa milloin asemassa olevaa levykettä on vaihdettu. SHAREn avustuksella se huomaa, mikäli käyttäjä on vaihtanut levykettä kesken sovelluksen ja estää levykkeen kirjanpidon sekoamisen. Kyky erottaa levykkeet toisistaan on erityisen tärkeä jatkossa, kun moniajoon pystyvät käyttöjärjestelmät yleistyvät. Niissä voi olla käynnissä useita eri ohjelmia yhtä aikaa, jolloin asemassa olevasta levykkeestä on pakko pitää tarkkaa kirjaa.

Entä mistä sarjanumero sitten saadaan? FORMAT-ohjelma ottaa sen ajasta ja päiväyksestä niin, että sarjanumeron ensimmäinen osa on alustusajan sekuntien, sekuntien sadasosien sekä kuukauden ja päivän järjestysnumeron summa. Toinen osa saadaan vastaavasti laskemalla yhteen vuosi sekä tunnit ja minuutit.

Jos levyke alustetaan esimerkiksi 6.12.92 klo 17:55:50.44, ensimmäinen osa saadaan laskemalla 0C06 (12 heksana ja 6 heksana, jenkkityyliin kuukausi ensin ja päivä sitten) + 322C (50 heksana ja 44 heksana) = 3E32. Jälkimmäinen osa saa arvon 07C8 (1992 heksana) + 1137 (17 heksana ja 55 heksana) eli 18FF ja koko sarjanumero on siten 3E32-18FF. Nämä laskutoimitukset on helpointa tehdä Windowsin mukana tulevalla tieteislaskimella.

Tekemällä lasku takaperin voidaan sarjanumerosta selvittää levykkeen alustusajankohta. Jos levylle on annettu alustuksen yhteydessä nimi, alustusajankohta saadaan selville helpomminkin, sillä CHKDSK katsoo sen nimestä:

```
C:\>CHKDSK B:

Volume KORPPU     created 06.12.1992 17.55
Volume Serial Number is 3E32-18FF

    730112 bytes total disk space
    730112 bytes available on disk
```

Voisiko sarjanumerolle keksiä mitään hyödyllistä käyttöä? Ohjelmoija voi käyttää sitä alkeellisena kopiosuojauksena siten, että ohjelma tarkistaa asemassa olevan levykkeen numeron ja kieltäytyy toimimasta, mikäli numerot eivät täsmää. Käytännössä menetelmä on huono, sillä levykkeestä saadaan kuitenkin täydellinen kopio DOS 3.3:n DISKCOPYllä eikä sarjanumeroon perustuvaa suojausta voi käyttää mikäli ohjelma halutaan kopioida kiintolevylle.

63

File creation error

Kopioitaessa tiedostoja levykkeelle saattaa käydä niin, että työ keskeytyy virheilmoitukseen File creation error (DOS 5.0:sta alkaen Cannot make directory entry), eikä levykkeelle pysty enää kopioimaan mitään. Näin siitä huolimatta, että levyllä näyttää vielä olevan runsaasti vapaata tilaa.

Virhe aiheutuu päähakemiston täyttymisestä. Jokaiselta levyltä on varattu alue päähakemistoa varten. Koska alue on kiinteän mittainen, se voi tulla täyteen ja sen jälkeen levylle ei enää pysty kopioimaan mitään, vaikka levykkeellä muuten olisi vielä tilaakin.

Päähakemiston koko määräytyy levyn kapasiteetin perusteella seuraavasti:

Levykkeen koko	Päähakemiston koko
360 ja 720 kt	112 nimeä
1,2, 1,44 ja 2,88 Mt	224 nimeä

Jos levykkeelle on annettu nimi, tiedostoja mahtuu yksi vähemmän. Jos levylle on lisäksi siirretty käyttöjärjestelmä, se varaa kolme nimipaikkaa (kaksi piilotettua tiedostoa + nimi).

Päähakemisto ei yleensä pääse loppumaan kesken, sillä ennen kuin kaikki nimipaikat on käytetty, levyke on tullut täyteen. Jos tiedostot ovat kuitenkin pieniä, päähakemisto saattaa täyttyä ennen vapaan tilan loppumista. Yksinkertainen laskutoimitus osoittaa, että ratkaiseva tiedostokoko on 720 kilon ja 1,44 megan levykkeillä 720/112=6,43 kilotavua. Jos tiedostojen keskipituus on tätä pienempi, päähakemisto uhkaa loppua ennen kuin kaikki tallennustila on käytetty.

Mitä siis tehdä jos päähakemisto täyttyy? Jos levykettä käytetään työtiedostojen — esimerkiksi pienten tekstien tai ohjelmapätkien arkistointiin — kannattaa levykkeelle perustaa alihakemistoja ja kopioida tiedostot niihin. Paitsi että eri nimiset hakemistot helpottavat tiedostojen

järjestämistä, ne poistavat myös päähakemiston loppumisesta aiheutuvan ongelman. Toisin kuin päähakemisto, alihakemisto on rakenteeltaan dynaaminen ja sen pituus kasvaa tarpeen mukaan.

Rajoitus koskee myös kiintolevyjä. Niillä päähakemiston koko on aina 512 nimeä kapasiteetista riippumatta. Tähän rajaan törmää tuskin koskaan, sillä kiintolevyn päähakemisto neuvotaan jättämään mahdollisimman tyhjäksi ja tiedostot neuvotaan tallentamaan alihakemistoihin. Näin onkin syytä tehdä, sillä muuten kiintolevylle mahtuisi vain 512 tiedostoa sen kapasiteetista riippumatta!

63 B

RAM-levyllä päähakemiston koko on oletusarvona vain 64 paikkaa, mikä täyttyy äkkiä. Tästä syystä RAM-levylle kannattaa joko perustaa alihakemisto, johon tiedostot sijoitetaan, tai sitten päähakemiston kokoa pitää kasvattaa DEVICE-rivillä annettavalla parametrilla. Esimerkiksi rivi

```
DEVICE=C:\DOS\RAMDRIVE.SYS  512 256 2048 /E
```

varaa käyttöön kahden megatavun RAM-levyn, ottaa käyttöön 512 tavun lohkot (sektorit) ja jättää päähakemistoon tilaa 256 tiedostonimelle.

64

Ohjelmallinen uudelleenkäynnistys (Boot)

Käyttöjärjestelmän lataaminen uudelleen eli englanniksi koneen "buuttaaminen" tapahtuu painamalla yhtä aikaa Ctrl-, Alt- ja Del-näppäimiä. Näitä näppäimiä ei voi tallentaa komentojonoon, koska yrityskin painaa niitä komentojonoa kirjoitettaessa johtaa välittömään buuttaamiseen ja kirjoitus katkeaa siihen.

Uusi lataus on kuitenkin mahdollista tehdä puhtaasti ohjelmallisin keinon. Helpoin ja elegantein ratkaisu on käyttää DOSin Debug-ohjelmaa ja putkikomentoa:

```
ECHO G=ffff:0 | DEBUG
```

ECHO-lause välittää komennon G=FFFF:0 Debugille, joka tulkitsee sen käskynä hypätä BIOS ROMissa olevaan käynnistysrutiiniin. Rivi sopii hyvin esimerkiksi komentojonon loppuun.

Niksissä tarvittava Debug-ohjelma puuttuu eräistä DOS-versioista. Microsoft ja IBM ovat katsoneet, että ohjelma on luonteeltaan niin tekninen ettei tavallisen käyttäjän kannata vaivata sillä päätään. Ellei Debuggia ole käytettävissä, joudutaan turvautumaan raakaan voimaan ja kirjoittamaan edellisen esimerkin konekieliset käskyt suoraan heksakoodina tiedostoon BOOT.COM:

```
EA 00 00 FF FF
```

Koodit ovat itse asiassa peräisin Vienna-viruksesta, joka satunnaisesti kirjoittaa niitä sotkemiensa ohjelmatiedostojen alkuun. Yritys käynnistää tällä tavalla sotkettu ohjelma johtaa koneen buuttaamiseen.

64 B

Hyppy BIOSin osoitteeseen FFFF:0000 suorittaa ns. kylmäkäynnistyksen. Siinä tarkistetaan muistin määrä ja kunto ennen kuin käyttöjärjestelmän lataus alkaa. Käynnistyminen tapahtuu nopeammin, jos se tehdään ns. lämpimänä käynnistyksenä (warm boot). Se saadaan aikaan sijoittamalla osoitteeseen 0400:0072 arvo 1234h ennen hyppykomennon antamista. WARMBOOT.COM-ohjelma luodaan seuraavasti:

```
N WARMBOOT.COM
E 0100 B8 40 00 8E C0 26 C7 06
E 0108 72 00 34 12 EA 00 00 FF
E 0110 FF
RCX
0011
W
Q
```

64 C

Jos käytetään välimuistiohjelman kirjoituspuskuria pitää varmistaa, että puskureissa ollut tieto on ehditty tallentaa levylle ennen kuin käynnistyskomento annetaan. Smartdrv-ohjelmassa tämä tapahtuu komennolla SMARTDRV /C.

Puskurien tyhjentäminen on erityisen tarpeellista silloin, kun kopioidaan uudet aloitustiedostot vanhojen päälle ja kone pakotetaan käynnistymään uudelleen. Ellei kirjoitusta ole ehditty viedä levylle asti mikro käynnistyy seuraavalla kerralla vanhoilla tiedostoilla!

Tämän vuoksi käynnistyksen suorittavan komentojonon loppuun pitää kirjoittaa

```
SMARTDRV /C
BOOT
```

65

Milloin kiintolevy pitää perusalustaa?

Innokkaan mikroharrastajan tuntee siitä, että hän suorittaa kiintolevylleen perusalustuksen eli low level formatoinnin vähintään kerran kuukaudessa. Se ei ole hakkeri eikä mikään, joka ei ole low level formatoinut omaa kiintolevyään!

Todellisuudessa perusalustus on tarpeen vain harvoissa tilanteissa ja nykyisin suosituilla IDE-levyillä se voi olla suorastaan vahingollinen.

Mitä low level format sitten tarkoittaa ja milloin sitä tarvitaan? Kyseessä on nimensä mukaisesti perusalustus, joka kirjoittaa uudelleen kiintolevyn pinnalla olevat magneettiset merkit. Käyttäjän tallentaman informaation lisäksi levyn pinnalla on monenlaisia kohdistus- ja osoitemerkkejä, joista kiintolevyn ohjauselektroniikka tunnistaa missä kohtaa levyä lukupää on ja osaa siirtää sitä uralta toiselle.

Perusalustus löytää levyn magneettiseen materiaaliin mahdollisesti tulleet virheet. Vuosien kuluessa levylle kirjoitetut osoitemerkit saattavat heikentyä ja magneettinen pintamateriaali kulua niin, että lukupää ei

enää pysty lukemaan tai kirjoittamaan levyä luotettavasti. Fyysiset tärähdykset, lukupään osumiset levyn pintaan ja äärimmäisen pienet siirtymät levyn tai lukuvarren akseleissa saattavat myös aiheuttaa virheitä.

Perusalustus pystyy korjaamaan osan näistä virheistä. Se kirjoittaa urat uudelleen magnetoimalla ne, jolloin lukupää osuu jälleen tarkasti uralle ja jos levyltä löytyy pysyvästi vioittuneita alueita, ne saadaan merkittyä levyn omaan vikakirjanpitoon. Levystä riippuen nämä vika-alueet saattavat näkyä DOSin FORMAT-ohjelmalle bad sectoreina tai sitten ne piilotetaan kokonaan. Tällöin perusalustusohjelma ohjaa vioittuneelle kohdalle osuvat luku/kirjoituspyynnöt toiseen kohtaan levyä, varasektoreille, jolloin CHKDSK ei näe levyllä yhtään bad sectoria. Joka tapauksessa vialliset alueet eristetään pois käytöstä.

Perusalustuksen saa tehdä vain vanhoille ST-506 levyasemille. Myös monissa ESDI- ja SCSI-levyissä on oma perusalustusohjelmansa, joka sijaitsee ohjainkortin ROM-alueella. Se käynnistetään DEBUGin avulla esimerkiksi kirjoittamalla

```
-g=C800:5
```

Tarkka osoite riippuu kuitenkin kortista ja se käy ilmi ohjainkortin mukana tulleista ohjeista. Viallisen aloitusosoitteen huomaa siitä, että kone jää täysin jumiin. Jos C800:5 ei toimi, kannattaa kokeilla osoitteita C800:0 ja C800:CCC.

IDE-levyillä perusalustuksen tekeminen vaatii valmistajan oman alustusohjelman, eikä yleisiä (geneerisiä) ohjelmia pidä käyttää. Ne kieltäytyvät joko kirjoittamasta levylle mitään tai sitten ne tuhoavat levyllä olevat ja lukupäätä ohjaavat merkit, jonka jälkeen levy on pilalla. Valmistajan omaa alustusohjelmaa ei useinkaan jaeta käyttäjille, koska sitä tarvitaan äärimmäisen harvoin.

Milloin perusalustus sitten on todella tarpeen? Lähinnä silloin, jos vuosia käytössä ollut kiintolevy alkaa antaa luku- tai kirjoitusvirheitä (Reading/writing error on drive C:, Sector not found, Bad allocation table tms.). Nämä ovat fyysisiä vikoja, joita ei voida korjata DOSin omilla työkaluilla (CHKDSK /F, RECOVER jne.)

Tärkeätä: Perusalustusta ei koskaan tarvita levyn tyhjentämiseen eikä myöskään viruksen tuhoamiseen. Tyhjentäminen käy yhtä tehokkaasti esimerkiksi PC Toolsin Destroy/Wipe tai Nortonin Wipedisk-ohjelmilla.

Silloinkin, kun perusalustus katsotaan tarpeelliseksi, kannattaa kokeilla mieluummin Spinrite-ohjelmaa. Se pystyy testaamaan levyn kunnon usein tehokkaammin ja vaarattomammin kuin perusalustusohjelma. Lisäksi se osaa säätää vanhoilla levyillä käytetyn lomituskertoimen tiedostoja tuhoamatta. Spinrite tunnistaa ESDI-, SCSI- ja IDE-levyt eikä tee niille vahinkoa.

65 B

Perusalustusta ei saa koskaan tehdä kylmälle levylle, vaan levyn pitää saada lämmetä toimintalämpötilaansa olemalla päällä vähintään pari tuntia ennen alustusta. Jos alustus tehdään kylmänä, levyn urat siirtyvät hieman levyn lämmitessä käytön aikana. Viimeistään tunnin, parin päälläolon jälkeen levy alkaa antaa virheilmoituksia.

Tällaisessa tapauksessa uusi perusalustus on ainoa vaihtoehto.

65 C

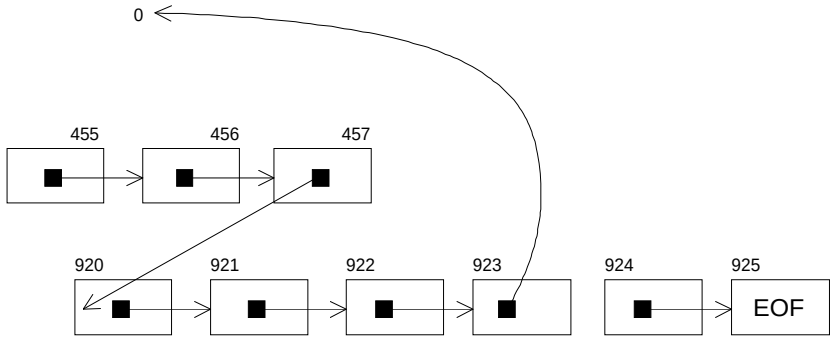
Fyysiset levyvirheet erottaa yleensä siitä, että virheilmoituksen jälkeen näkyy DOSin ns. kriittisen virheen käsittelijän kysymys

```
Abort, Retry, Ignore, Fail?
```

Jos kiintolevy antaa tällaisia virheitä, vika on joko levyn ohjaimessa tai itse levyssä, jolloin perusalustus tai osien vaihto on ainoa tapa saada järjestelmä taas kuntoon. Tästä säännöstä on kuitenkin yksi poikkeus: jos tiedoston käsittelyssä (esimerkiksi luettaessa tai kopioitaessa) tulostuu ilmoitus

```
Sector not found reading drive C
Abort, Retry, Ignore, Fail?
```

on mahdollista, että levyssä ei olekaan fyysistä vikaa vaan ainoastaan kirjanpito käsiteltävän tiedoston kohdalta on vioittunut. Tiedoston



Tiedoston sijainnista kertovaan FATiin voi tulla looginen vika, joka saa FAT-alkioiden ketjun katkeamaan ja osoittamaan nollaa. Virheellisen tiedoston käyttö saattaa pysähtyä virheeseen "Sector not found", joka näyttää erehdyttävästi vakavalta fyysiseltä levyvirheeltä.

varausketjuun on jostain syystä eksynyt arvo nolla, joka saa käyttöjärjestelmän yrittämään levyn lukua kohdasta, joka ei varmasti ole mahdollista. Tämä tuottaa huolestuttavan, aivan laitevialta näyttävän ilmoituksen.

Miten sitten erottaa, onko kyseessä perusalustusta vaativa levyvirhe vai ainoastaan kirjanpidossa oleva looginen virhe? Varausketjussa oleva nolla tuottaa yleensä CHKDSK-ajossa muunkin virheen, koska se katkaisee tiedostolle varatun ketjun kahteen osaan eikä tiedostolle varattujen varausyksiköiden määrä vastaa hakemistolistauksessa näkyvää tiedostopituutta. CHKDSK raportoi ensimmäisen virheen Lost clustereina ja huomauttaa toisesta, että "Has invalid allocation unit, file truncated". Jos nolla-alkio on kuitenkin ketjun viimeisenä, virheilmoituksia ei välttämättä tulostu. Tiedosto saattaa jopa kopioitua, mutta kopio ei kuitenkaan toimi.

Jos kyse on varausketjussa olevasta nollasta, siitä pääsee eroon yksinkertaisesti poistamalla virheilmoituksia tuottavan tiedoston ja ajamalla sen jälkeen CHKDSK /F.

66

Kiintolevyn alustuksen voi keskeyttää

Kun FORMAT-ohjelma alustaa kiintolevyä, levyn merkkivalo palaa tasaisesti ja ruudulla näkyy tieto siitä, miten alustaminen etenee. Vanhoissa DOS-versioissa näkyy käsiteltävän sylinterin numero, uudemmissa sen tilalla on etenemistä kuvaava prosenttiluku. Moni käyttäjä kuvittelee, että FORMAT käy tällöin kirjoittamassa levyn täyteen nollaa ja tuhoaa samalla kaikki vanhat tiedot.

Näin ei kuitenkaan tapahdu. Levyn valon palaessa FORMAT ei suinkaan kirjoita levylle vaan se *lukee* sitä. Lukemalla levy alusta loppuun FORMAT varmistaa, ettei levylle ole tullut uusia vika-alueita. Jos niitä löytyy, FORMAT pistää vikapaikat muistiinsa ja kirjoittaa ne lopuksi bad sectoreina FATiin. Samalla FORMAT tyhjentää päähakemiston ja levyn käytöstä kertovan kartan, jolloin levy näyttää tyhjältä.

Koska varsinaisiin tiedostoihin ei kosketa mitenkään, niitä voi tutkia ja osin jopa palauttaakin alustuksen jälkeen. Tämä kannattaa pitää mielessä silloin, kun on esimerkiksi myynyt mikronsa seuraavalle käyttäjälle ja haluaa tyhjentää sen vanhoista tiedostoista.

Jos siis huomaat jonain päivänä alustavasi väärää konetta, peli ei ole vielä menetetty. Jos ehdit painaa Ctrl+Alt+Del ennen kuin FORMATin etenemisestä kertova prosenttiluku on ehtinyt liian lähelle sataa, kone käynnistyy uudelleen ja huomaat, että levy on aivan entisensä. Mitään vahinkoa ei ole tapahtunut, koska olet ehtinyt keskeyttää FORMATin ennen kuin se on ryhtynyt kirjanpitoalueen tyhjentämiseen.

Kokeile vaikka.

67

Levykkeiden varma alustaminen

Toisin kuin kiintolevyillä, levykkeillä FORMAT suorittaa fyysisen alustuksen eli se kirjoittaa levykkeen urat uudelleen ja samalla tuhoaa niillä

kaiken aiemmin olleen tiedon. Jos levykkeen alustaminen on ehtinyt alkaa, sitä on turha keskeyttää. Tiedostoja ei kuitenkaan saa pelastettua.

DOS 5.0:sta alkaen FORMATin toimintaperiaatetta kuitenkin muutettiin niin, että jos levyke on jo kerran alustettu, sille tehdään kiintolevyn tapaan ainoastaan looginen alustus. FORMAT tyhjentää päähakemiston ja FATin, jonka jälkeen levyke näyttää tyhjältä. Levykkeellä aiemmin olleita tiedostoja voidaan kuitenkin yhä tutkia lukemalla niitä suoraan levyn pinnalta.

Tämä aiheuttaa selvän tietoturvariskin silloin, kun tiedostoja toimitetaan levykkeillä toisille käyttäjille. Kukapa enää jälkeenpäin muistaisi, mitä luottamuksellisia tiedostoja levykkeillä on aikaisemmin ollut. Ties mitä salaisuuksia utelias vastaanottaja pystyy kaivamaan levykkeeltä esiin.

Vielä DOS 5.0:n jälkeenkin levyke saadaan todella tyhjennettyä, mutta apuun tarvitaan /U-valitsinta (unconditional). Komento

```
FORMAT A: /U
```

takaa, että levy todella pyyhitään tyhjäksi alustamisen aikana ja ettei vastaanottaja enää pysty näkemään mitä tiedostoja sillä on aikaisemmin ollut.

67 B

DOS 5.0:sta alkaen käyttöjärjestelmä tutkii ennen alustuksen aloittamista, onko levyä käytetty. Jos levyllä on jo tiedostoja, DOS ajaa ensin MIRRORin ja aloittaa varsinaisen alustamisen vasta tämän jälkeen.

Mikäli alustettava levyke on jo ollut käytössä ja se on lähes täynnä, MIRROR-aputiedostolle ei ole tilaa ja siksi FORMAT antaa varoituksen

```
Drive A: error. Insufficient space for the MIRROR
image file. There was an error creating the format
recovery file.
This disk cannot be unformatted.
Proceed with Format (Y/N)?
```

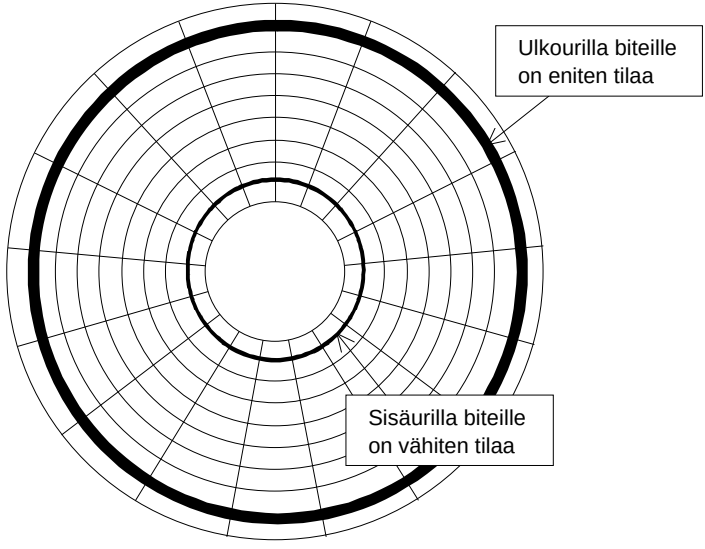
Levykkeen voi tässäkin tapauksessa alustaa, mutta UNFORMAT-komento ei toimi MIRROR-tiedoston puuttuessa.

Jos käytetään /U-valitsinta, varoitusta ei tule koska MIRRORia ei silloin ajeta.

68

Tallenna levykkeelle tärkeimmät tiedostot ensiksi

Levykkeen sisällä tiedostot tallennetaan reikäleipää muistuttavalle magneettiselle levylle. Loogisesti levy jakautuu uriin, joista sektorit leikkaavat lohkoja. Englanninkielessä näitä lohkoja kutsutaan harhaanjohtavasti sanalla sector. Koska sektorien määrä on sama kaikilla urilla, uloimmista lohkoista tulee selvästi pidempiä kuin sisimmistä.



Uloimmat urat ovat turvallisimpia, koska niissä biteillä on eniten tilaa. Tärkeän tiedoston voi varmuuden vuoksi tallentaa kahdesti, ensin levyn alkuun ja toisen kerran sen loppuun. Näin kopiot tulevat mahdollisimman kauaksi toisistaan.

Vaikka ulkoreunan lohkot ovatkin pitempiä kuin sisäreunan lohkot, niihin tallennetaan aina sama määrä tietoa. Sisimmillä urilla bitit ovat vain lähempänä toisiaan kuin uloimmilla urilla. Ja mitä tiiviimpään bitit on pakattu, sitä herkemmäksi ja virhealttiimmaksi niiden käsittely muuttuu.

Kun tyhjälle levykkeelle aletaan tallentaa tiedostoja, DOS täyttää levyä järjestyksessä uloimmilta urilta alkaen. Tärkeimmät tiedostot kannattaa siksi kopioida levylle ensiksi, jotta ne tallentuvat turvallisimmille ulkourille. Tästä syystä myös kirjanpitoalueet sijaitsevat levyn ulkoreunalla eikä keskellä, mikä olisi nopeuden kannalta parempi paikka.

Ja jos tiedosto on erityisen tärkeä, se kannattaa kopioida levylle vielä toiseenkin kertaan silloin, kun levy on jo lähes täynnä. Näin tiedoston molemmat kappaleet tulevat eri puolille levyä ja jos levyyn tulee esimerkiksi naarmu tai muu fyysinen vika, ainakin toisen tiedostoista pitäisi olla luettavissa.

69

Älä käytä työtiedostoja suoraan levykkeiltä

Takavuosina, kun kiintolevyt olivat harvinaisia ja monissa koneissa oli vain A: ja B: levykeasemat, oli tapana sijoittaa työtiedostot levykkeelle ja pitää sitä B: asemassa. A: asemaan laitettiin ohjelmalevyke.

Kiintolevyjen tultua työtiedostot voitiin kopioida sille, jonka jälkeen levykkeitä ei enää tarvittu muuhun kuin tiedostojen varmistamiseen. Eikä niitä muuhun pitäisi käyttääkään.

Syy tähän on DOSin kyvyssä — tai pikemminkin kyvyttömyydessä — havaita, milloin asemassa olevaa levykettä on vaihdettu. Ja jos sovellus ei huomaa, että levyke on vaihtunut, se saattaa kirjoittaa uudelle levykkeelle edellisen levykkeen kirjanpidon mukaisesti, jolloin sen tiedostot tuhoutuvat. Levykettä ei saa vaihtaa, jos sovelluksella on auki olevia tiedostoja — ja sitähän käyttäjän on mahdotonta tietää.

DOS 4.0:sta lähtien levykkeille syntyy alustuksen yhteydessä yksilöllinen sarjanumero. Jos vielä SHARE on ladattu muistiin, DOS huomaa

heti ensimmäisellä luku- tai kirjoituskomennolla että levykettä on vaihdettu ja antaa varoituksen.

```
Invalid disk change writing drive A
Please insert volume TYOT serial AB83-91FF
```

Virheilmoituksella DOS kertoo, mikä levyke asemassa pitäisi olla. Vahinko voi silti tapahtua jos SHAREa ei ole ladattu.

Toinen tilanne, joka DOSissa johtaa levykkeen vahingoittumiseen, on seuraava:

Käyttäjä haluaa kopioida hakemiston tiedostot levykkeelle ja antaa tätä varten komennon COPY *.* A: . Levyke on kuitenkin kirjoitussuojattu, joten kopiointi pysähtyy virheilmoitukseen

```
Write protect error writing drive A
Abort, Retry, Fail?
```

Tällöin käyttäjä vaihtaa asemaan toisen, suojaamattoman levykkeen ja painaa R:ää, jolloin DOS jatkaa kopiointia. Se toimii kuitenkin yhä ensimmäiseltä levykkeeltä lukemansa kirjanpidon mukaisesti joten COPY kirjoittaa levykkeellä mahdollisesti olevien tiedostojen päälle ja sotkee kirjanpidon.

Levykkeen kohtelu riippuu jossain käytettävästä ohjelmasta tai komennosta. Esimerkiksi XCOPY tarkistaa levykkeen ennen kopioinnin aloittamista ja antaa varoituksen, mutta tavallinen COPY ei. Jos SHARE on ladattu, molemmat komennot huomaavat levykkeen vaihdon — kunhan vain molemmilla levykkeillä on ainakin eri nimet erottamassa niitä toisistaan.

Levykkeiden käyttö voi tuottaa ongelmia SHAREsta huolimatta. Jos esimerkiksi Word for Windowsissa halutaan upottaa levykkeellä oleva tekstitiedosto avoinna olevaan dokumenttiin, tiedosto kannattaa kopioida kiintolevylle ennen lisäyskomennon antamista. Toinen vaihtoehto on avata tekstitiedosto vaikkapa Notepadillä, sulkea tiedosto ja lisätä teksti sen jälkeen Wordiin leikepöydän kautta.

Jos käytetään suoraa lisäyskomentoa Insert/File, Word luo ainoastaan linkin upotettavaan tiedostoon ja jättää tiedoston auki. Käyttäjä luulee, että levyke on tehnyt tehtävänsä ja poistaa sen asemasta. Kun Wordissä aikanaan annetaan tekstin tallennuskomento, ohjelma yrittää lukea

levykkeellä olevaa tiedostoa ja kun tämä ei onnistu, tulostuu levyvirheestä kertova varoitus. Sen seurauksena Wordillä tehty varsinainen tekstitiedosto saattaa mennä sekaisin.

Käyttäjän on mahdotonta tietää, milloin sovellus todella sulkee avaamansa tiedoston. Kun lisättävä tiedosto kopioidaan ennen lisäyskäskyn antamista kiintolevylle varmistetaan, että tiedosto on koko ajan Wordin saatavilla.

Kaikilta näiltä ongelmilta välttyy, kun ei käsittele työtiedostoja suoraan levykkeiltä. Jos työtiedostoja halutaan säilyttää levykkeillä, on parempi kopioida ne käytön ajaksi kiintolevylle ja sen jälkeen takaisin levykkeille.

70

Kannattaako levyn pakkausohjelmia käyttää?

Idea kiintolevyn pakkaamisesta lennossa niin, että se näyttää isommalta, ei ole uusi. Vuosien varrella tehtävää varten on kehitetty sekä ohjelmallisia ratkaisuja että lisäkortteja. Kuitenkin vasta 1991 Stacker räjäytti potin ja toi ajonaikaisen levypakkauksen kaikkien tietoisuuteen.

Lyhyessä ajassa Stacker ja sen kanssa kilpailevat ohjelmat levisivät kulovalkean tavoin tilanpuutteesta kärsivien käyttäjien keskuuteen ja toivat kaivattua helpotusta varsinkin Windows-käyttäjille. Microsoft antoi asialle epävirallisen hyväksyntänsä kun se lisäsi DOS 6.0:aan oman versionsa pakkauksesta.

Levyn pakkaus on houkutteleva ajatus — kukapa ei haluaisi levyasemansa näyttävän entistä isommalta? Mutta ilmaista lounasta ei ole. Pakatun levyn käytössä on omat haittansa:

- Pakattu levyasema toimii 10-15 prosenttia hitaammin, koska tiedostoja pitää koko ajan pakata ja purkaa.
- Pakkauksen suorittava ohjelma vie 30-50 kilotavua keskusmuistia. Varsinkin 286-koneissa, joissa ei ole ylämuistia, pakkauksen vaatima muistintarve on liian suuri. Jos pakkausohjelma lada-

taan muistiin, sovellukset (ja varsinkin isot pelit!) eivät enää toimi.

- Pakattu levy on alttiimpi erilaisille ongelmille, kuten kirjanpidon sekoamisille. Ja jos vikoja sattuu, korjausohjelmat eivät yleensä osaa käsitellä pakattuja levyjä.
- Vapaan tilan arvioiminen on hankalaa, koska levylle vielä mahtuva tietomäärä riippuu tiedon sisällöstä.
- Ohjelmat, jotka vaativat levyltä yhtenäisen alueen (kuten Windowsin kiinteä virtuaalimuisti), eivät toimi pakatulla levyllä.
- Muut käyttöjärjestelmät, kuten OS/2 tai Windows NT, eivät yleensä pysty käsittelemään pakatulla levyllä olevia tiedostoja.
- Jos levyllä säilytetään jo valmiiksi pakattuja tiedostoja (kuten GIF-kuvia tai ARJ/ZIP/LZH-paketteja) pakkauksesta ei ole mitään hyötyä.

Levyn pakkaus muistuttaa velan ottamista. Se saa aseman näyttämään isommalta, mutta ei ratkaise tilaongelmaa. Ennen pitkää levy on jälleen täynnä eikä lisälainaa enää saa.

Jos et siis ehdottomasti tarvitse levyn pakkausta, älä käytä sitä. Opettele mieluummin poistamaan tarpeettomat tiedostot ajoissa, jotta tila ei pääse loppumaan.

Ja jos tarvitset lisää levytilaa, tarkista ensin riittäisivätkö rahasi uuden levyn hankintaan.

Näistä haittapuolista huolimatta pakkauksen käyttö saattaa olla suositeltavaa muistikirjamikroissa. Niissä levyasema saattaa jo lähtötilanteessa olla aivan liian pieni (20-40 megatavua), eikä sitä pysty mitenkään laajentamaan.

Silloin pakkausohjelman käyttö saattaa olla paitsi perusteltua myös ainoa tapa saada tiedostot mahtumaan.

71

Kannattaako levyn pirstoutumista poistaa?

Innokas PC-harrastaja järjestää levynsä tiedostot vähintään kerran viikossa peräkkäin PC Toolsin Compressia tai Nortonin Speed Diskiä käyttäen. Levy näyttää niin paljon siistimmältä, kun tiedostoja kuvaavat merkit ovat ohjelman järjestäminä peräkkäin!

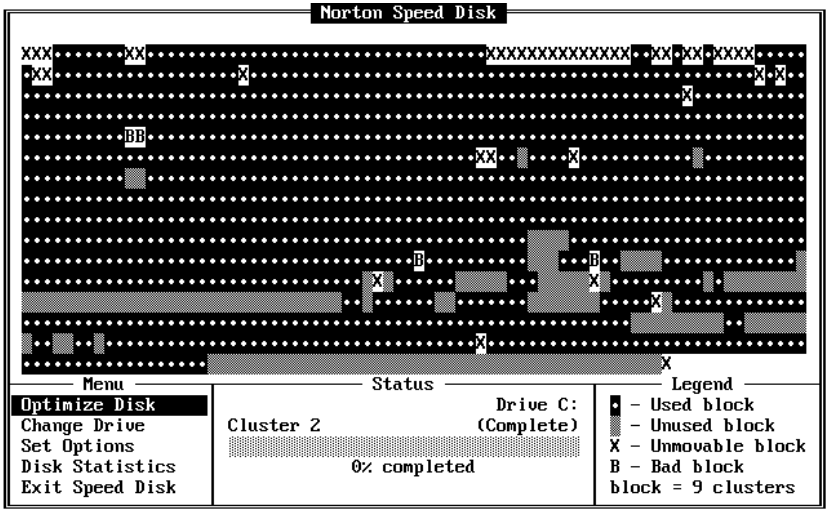
Tiedostojen järjestämiselle (unfragment) on olemassa omat syynsä. Kun levyllä olevia tiedostoja poistetaan, uusia kopioidaan tilalle ja vanhoja tiedostoja pidennetään, DOSin on otettava käyttöön muilta tiedostoilta vapautuneita alueita. Aikaa myöten iso tiedosto saattaa olla levyllä useana pienenä palasena siellä täällä. Vastaavasti levyn vapaa tila on jakautunut useaan pieneen alueeseen eri puolille levyä.

Tätä ilmiötä kutsutaan pirstoutumiseksi eli fragmentoitumiseksi. Se ei estä tiedostojen käyttöä, sillä DOS osaa kyllä lukea ja kirjoittaa tiedostoja olivatpa ne sitten pirstoutuneita tai ei. Pirstoutuminen hidastaa kuitenkin tiedostojen käsittelyä ja ohjelmien lataamista, sillä lukupää joutuu hakemaan tiedoston osia eri puolilta levyä.

Vanhoilla ja hitailla levyasemilla pirstoutuminen saattaa näkyä selvästikin. Vanhat ATK-ammattilaiset kertovat tarinoita 70-luvulta, jolloin levy-yksiköt olivat korkeita laatikoita. Silloin tiedostojen pirstoutumisen saattoi päätellä siitä, miten pahasti levyasema tärisi käytön aikana.

Tuo kaikki on kuitenkin historiaa. Nykyiset nopeat levyasemat lukevat tiedoston lähes ajatuksen nopeudella olipa se sitten pirstoutunut tai ei.

Tarkastellaanpa asiaa levyllä, jonka hakuaika on 20 millisekuntia (0,02 sekuntia) ja tiedonsiirtonopeus 500 kilotavua sekunnissa. Lisäksi pitää ottaa huomioon levyjen pyörimisestä aiheutuva viive eli latenssi. Siitä aiheutuu keskimäärin puoleen kierähdykseen kuluva lisäodotus oli 8 millisekuntia. Siirrettävänä on tiedosto, jonka koko on 500 kilotavua. Mikäli tiedosto on levyllä peräkkäin, riittää sen etsiminen (20 millisekuntia) ja levyn pyörähtäminen oikeaan kohtaan (8 millisekuntia). Sen jälkeen tiedoston siirtäminen kestää yhden sekunnin, jolloin kokonaisaika on 1,028 sekuntia.



Järjestelyohjelman näyttö kertoo vapaan tilan hajaantuneen eri puolille levyä. Levy on "reikiintynyt". Ohjelma järjestää tiedostot peräkkäin ja kokoaa vapaan tilan yhteen, mutta tällä toimenpiteellä ei ole juurikaan vaikutusta mikron toimintaan.

Kun tiedosto on pirstoutunut vaikkapa kymmeneen osaan, joudutaan tekemään kymmenen levyhakua ja odotusta eli yhteensä 10*(20+8) = 0,28 sekuntia. Kokonaissiirtoaika kasvaa siten 1,28 sekuntiin eli 0,252 sekunnilla. Sekunnin neljäsosan lisäviive on niin vähäinen, ettei sitä huomaa ilman tarkkaa sekuntikelloa. Toisaalta voi ajatella, että levyoperaatiot hidastuvat noin 25 prosenttia millä saattaa olla merkitystä silloin, kun levyä luetaan ja kirjoitetaan jatkuvasti esimerkiksi lähiverkon serverissä.

Jos käytetään välimuistia, pirstoutumisen merkitys on entistä vähäisempi. Koska se pitää muistissaan aiemmin luettuja kohtia levystä, lukupään liike vähenee eikä pirstoutumisen vaikutus pääse näkymään.

Tiedostojen järjestäminen ei säästä käyttäjän aikaa vaikka tiedostojen lukeminen hieman nopeutuisikin, sillä tiedostojen järjestämiseen kuluu nopealtakin ohjelmalta useita minuutteja — jopa puoli tuntia — ja tarvitaan todella monia levyhakuja, ennen kuin tämä aika on saatu säästönä takaisin.

Joku toinen on perustellut tiedostojen järjestämistä sillä, että levyaseman lukupää kuluisi vähemmän kun se joutuisi tekemään vähemmän edestakaista liikettä tiedostoja lukiessaan. Käytännössä asia on kuitenkin päinvastoin: tiedostojen järjestäminen peräkkäin vaatii niin monta edestakaista liikettä, että lukupää kuluu enemmän kuin satunnaisesti hajallaan olevia tiedostoja käsiteltäessä!

Viimeinen seikka, joka vähentää tiedostojen järjestelyn mielekkyyttä, on prosessin vaarallisuus. Kun ohjelma järjestelee tiedostoja levyllä, se joutuu lukemaan ja kirjoittamaan suoraan kirjanpitoalueiden (FAT) päälle. Pienikin virhe ohjelmassa tai yhteensopivuudessa riittää sotkemaan kirjanpidon ja tekemään tiedostot käyttökelvottomiksi. Miksi siis ottaa riskiä? Järjestelyohjelmaan kulutetut rahat kannattaa käyttää johonkin hyödyllisempään.

Kaikesta huolimatta tiedostojen järjestämisellä on hyvätkin puolensa. Esimerkiksi Windowsin kiinteä virtuaalimuisti vaatii levyltä yhtenäisen vapaan tilan, jolloin tiedostoja voi olla pakkokin järjestää — tai sitten on tyytyminen vain pieneen virtuaalimuistin aputiedostoon.

Toinen tilanne, jossa tiedostojen peräkkäisyydestä on selvää hyötyä, liittyy vahingossa poistettujen tiedostojen palauttamiseen. Mitä enemmän tiedosto on pirstoutunut, sitä hankalampaa sen palauttaminen on. Levyllä peräkkäin olevan tiedoston palauttaminen on sitä vastoin erittäin helppoa.

Jos käytetään DOS 5.0:sta alkaen tulevaa MIRROR-ohjelmaa tai jotakin sen kaupallista vastinetta, pirstoutumisella ei ole merkitystä, sillä ohjelmat muistavat missä poistetun tiedoston eri palaset ovat sijainneet ja osaavat palauttaa tiedostot luotettavasti.

72

Yksittäisen tiedoston järjestäminen

Vaikka edellä todettiin, ettei tiedostojen pirstoutuminen yleensä muodostu ongelmaksi asti, erikoistapauksissa näin voi käydä. Lisäksi tiedostojen pirstoutumisen selvittäminen voi joskus antaa mielenkiintoisia tuloksia.

CHKDSK osaa kertoa, mitkä oletushakemiston tiedostot ovat pirstoutuneet ja moneenko palaseen tiedostot ovat jakautuneet. Valitettavasti CHKDSK osaa tutkia vain yhden hakemiston eikä koko levyä kerrallaan. Oletushakemiston tutkiminen tapahtuu komennolla

```
CHKDSK *.*
```

Jos kaikki tiedostot ovat peräkkäisiä, CHKDSK ilmoittaa

```
All specified files are contiguous.
```

Muutoin se kertoo pirstoutuneiden tiedostojen nimet ja osien määrän:

```
D:\W2\M-KATSAU.DOC Contains 4 non-contiguous blocks
D:\W2\ICL-PEN.TXT Contains 3 non-contiguous blocks
D:\W2\ANNELLE.DOC Contains 2 non-contiguous blocks
D:\W2\WFWG.DOC Contains 9 non-contiguous blocks
D:\W2\PALS0992.DOC Contains 3 non-contiguous blocks
D:\W2\PPOINT3.DOC Contains 3 non-contiguous blocks
```

Eniten pirstoutuvat tiedostot, joita kasvatetaan pienissä paloissa. Tällaisia tiedostoja ovat esimerkiksi kasvavat tietokannat sekä tekstitiedostot. Mitä useampia välitallennuksia, sitä enemmän tiedosto on pirstoutunut. Esimerkiksi tämän kirjan teksti näytti eräässä vaiheessa tällaiselta:

```
D:\KIRJA\NIKS.DOC Contains 31 non-contiguous blocks
```

Vaikka tiedosto olikin levyllä 31:nä erillisenä palana, pirstoutuminen ei mitenkään haitannut tekstin käsittelyä eikä havaittavasti edes hidastanut sitä. Jos pirstoutumisen kuitenkin haluaa poistaa, on hyvä tietää asiaan liittyvä niksi: kopiointi.

Kun pahasti pirstoutunut tiedosto kopioidaan hetkeksi toiselle nimelle, alkuperäinen poistetaan ja tehty kopio nimetään takaisin alkuperäisen tiedoston nimelle, havaitaan että uusi tiedosto on joko täysin yhtenäinen tai ainakin sen osien määrä on selvästi vähentynyt. Jos ensimmäinen kopiointi ei vielä tuota haluttua tulosta, kannattaa kokeilla aputiedoston luomista toiseen asemaan. Ja jos useat hakemiston tiedostot ovat pirstoutuneet, menettelyn voi toistaa koko hakemistolle.

Miksi sitten pelkkä tiedoston kopiointi riittää vähentämään sen pirstoutumista? Siksi, että kun DOS etsii tilaa uudelle tiedostolle, se ei suinkaan ala kirjoittaa sitä ensimmäiseen vapaaseen kohtaan mitä levyltä löytyy. Sen sijaan DOS käyttää yksinkertaista heuristiikkaa, jolla se pyrkii löytämään joko tarpeeksi ison yhtenäisen alueen tai ainakin jakamaan tiedoston niin, että paloja tarvitaan mahdollisimman vähän. Heuristiikka ei ole kovin monimutkainen, mutta useimmiten riittävä.

Mitä enemmän levyllä on vapaata tilaa, sitä helpompi DOSin on sijoittaa tiedostot levylle. Kun vapaa tila alkaa loppua, pirstoutumisen määrä kasvaa nopeasti. Tästä saadaan yksinkertainen sääntö: älä koskaan päästä levyä aivan täyteen vaan huolehdi aina siitä, että vähintään 15 prosenttia levytilasta säilyy vapaana.

Tämä vähentää pirstoutumista niin, ettei erillistä järjestysohjelmaa tarvita koskaan.

73

Vieraskieliset virheilmoitukset

Vaikka kiintolevylle olisi asennettu englanninkielinen DOS, se saattaa joskus antaa suomen- tai jonkin muun kielisiä virheilmoituksia. Eräs mikronkäyttäjä lähetti jopa huolestuneen kirjeen minulle ja epäili virusta, kun hänen koneensa eräänä aamuna käynnistymisen sijaan tulostikin ruudulle tekstin

```
Diskfejl eller manglende DOS.
Anvend DOS Disk og tryk på Enter
```

Miten oli mahdollista, että kone antoi tanskankielisen virheilmoituksen vaikka koneessa oli englanninkielinen DOS-versio?

Kuvaruudulle tulostuva virheilmoitus on peräisin asemaan unohtuneelta levykkeeltä. Jokaisella levykkeellä on pieni ohjelma, joka yrittää aloittaa käyttöjärjestelmän latauksen ja antaa virheilmoituksen, jos käyttöjärjestelmää ei löydy.

Ilmoitus on tuttu monelle PC-käyttäjälle:

```
Non-System disk or disk error
Replace and press any key when ready
```

FORMAT-ohjelma kirjoittaa tämän ohjelman ja siihen liittyvät virheilmoitukset levylle alustaessaan sitä. Ilmoitus kirjoitetaan jokaiselle levykkeelle eikä sillä ole mitään tekemistä käyttöjärjestelmän kanssa. Se syntyy levykkeelle, vaikkei sille siirrettäisikään käyttöjärjestelmää /S-valitsimella.

Eri maiden kansallistetut DOS-versiot kirjoittavat ilmoituksen tietenkin omalla kielellään ja jos levyke on aikanaan alustettu tanskalaisella DOSilla, sen FORMAT kirjoittaa käynnistyslohkon ohjelmaan virheilmoituksen omalla kielellään.

Englanninkielisenkin virheilmoituksen sanamuoto voi vaihdella. Esimerkiksi PC Toolsin alustusohjelma kirjoittaa käynnistyslohkoon DOSin omaa FORMATia selkeämmän tekstin

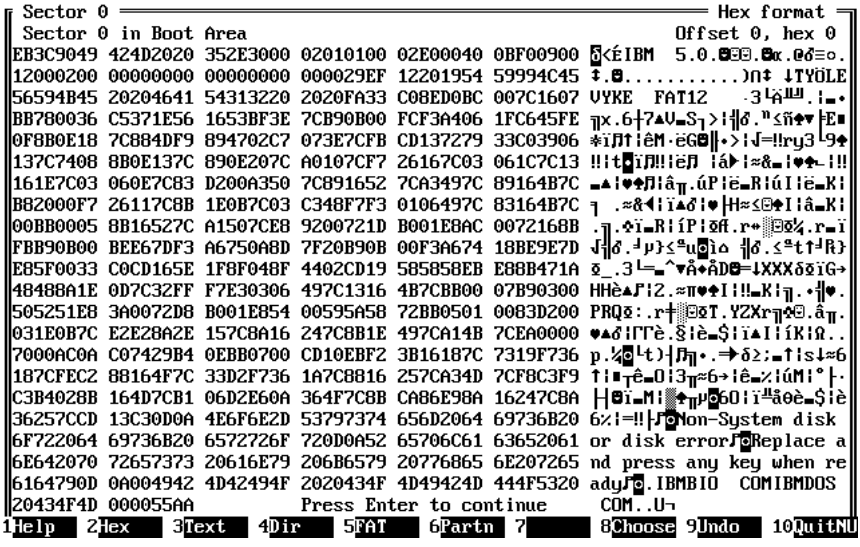
```
This disk can't boot. It was
formatted without the /S (system) option.

To make it bootable, use the DOS utility SYS x:

Change disks & press  a key.
```

Miksi virheilmoitus sitten tulee suoraan levykkeeltä eikä koneessa olevasta käyttöjärjestelmästä? Koska käynnistyksen alkaessa mitään käyttöjärjestelmää ei vielä ole muistissa ainoaksi mahdollisuudeksi jää saada virheilmoitukset suoraan levyltä.

74

Millä DOS-versiolla levyke on alustettu?

Kun levyke alustetaan, FORMAT-ohjelma kirjoittaa sen käynnistyslohkoon oman tunnuksensa. Tämä tunnus syntyy jokaiselle levylle riippumatta siitä, onko sillä käyttöjärjestelmä vai ei.



Käynnistyslohkon (boot sector) alussa näkyy FORMAT-ohjelman kirjoittama tunnus. Se kertoo, millä DOS-versiolla levyke on alustettu. Lohkon lopussa näkyvät selväkieliset virheilmoitukset. Ne tulostuvat ruudulle jos alkulataus (buuttaus) ei onnistu.

DOS-tunnus sijaitsee lähes heti käynnistyslohkon alussa. Aiemmin, kun mikroissa vielä käytettiin valmistajakohtaisesti sovitettuja MS-DOS -versioita, tunnuksesta saattoi nähdä millä koneella levyke oli alustettu. Esimerkiksi Nokian MS-DOS 3.2 kirjoitti tunnukseksi "NOKIA3.2" ja Toshiban DOS 4.0 "T V4.00". Sittemmin laitekohtaiset erot ovat pienentyneet niin, että lähes kaikki käyttävät joko standardia MS-DOSia tai PC-DOSia tai jotakin apuohjelmaa, jolla on oma tunnuksensa (PC Tools "PCFormat" ja Norton Utilities "IBM PCNI").

Tunnuksesta näkee myös onko levyke alustettu DOSilla vai OS/2:lla. Vaikka niiden käyttämät levyformaatit ovat muuten samat, OS/2:n FORMAT kirjoittaa versionumeroksi vähintään 10. Numero 20.0 vastaa OS/2-versiota 2.0 ja niin edelleen.

DOS-tunnuksen tutkiminen voi joskus antaa mielenkiintoisiakin tuloksia. Esimerkiksi Microsoftilta tulleissa ohjelmalevykkeissä alkoi lukea MSDOS 5.0 jo kauan ennen kuin Microsoft julkisti DOS 5.0:n —

merkki siitä, että Microsoft suoritti uuden DOS-version testausta talon sisällä ennen sen julkistamista.

Valmisohjelmalevykkeistä voi myös päätellä, millaista DOS-versiota käyttävät Lotus, WordPerfect, Borland ja muut tunnetut ohjelmatalot.

Tiedostot

75

Kopiointi ja tiedostomääreet

DOSin kopiointikäsky ja XCOPY-apuohjelma suhtautuvat tiedostomääreihin tavalla, joka saattaa joskus tuottaa yllätyksiä. Vaikka kopioitava tiedosto olisi suojattu lukumääreellä, kopiotiedosto ei enää ole vaan se pitää suojata erikseen ATTRIB +R komennolla kopioinnin jälkeen.

Arkistointimääre käyttäytyy täsmälleen päinvastoin. Vaikka kopioitavassa tiedostossa ei olisi sitä, kopiotiedosto saa sen. Tämä onkin luonnollista, sillä kopioinnin tuloksena syntynyt tiedosto on "uusi" ja jos esimerkiksi varmuuskopiointirutiini nojaa arkistointimääreen käyttöön, määre kertoo että tiedosto pitää ottaa mukaan seuraavaan varmistusajoon.

Piilotus- tai järjestelmämääreellä suojattuja tiedostoja ei voi kopioida, sillä sen paremmin COPY kuin XCOPYkään ei näe niitä.

75 B

Mielenkiintoista kyllä, TYPE ei piittaa piilotuksista. Jos vain tietää tiedoston nimen, sen voi tulostaa ruudulle TYPE-komennolla normaaliin tapaan.

76

Piilotettujen tiedostojen tarkistaminen

Jos on aihetta epäillä että hakemistossa on piilotettuja tiedostoja, niiden nimet ja muut kirjanpitotiedot nähdään helposti kirjoittamalla DIR /AH

tai DIR /AS. Ensimmäinen komento näyttää piilotetut tiedostot, jälkimmäinen järjestelmätiedostot jotka myös ovat näkymättömiä. DIR /A-käsky toimii kuitenkin vasta DOS 5.0:sta alkaen joten vanhemmissa DOS-versioissa on keksittävä jotakin muuta.

Eräs keino on poistaa kaikki hakemiston tiedostot komennolla DEL *.* ja yrittää sen jälkeen poistaa itse hakemisto RD-komennolla. Koska hakemisto voidaan poistaa vain tyhjänä, RD antaa virheilmoituksen mikäli hakemistoon on vielä jäänyt piilotettuja tiedostoja. Se ei kuitenkaan kerro niiden nimiä.

Nimet saadaan selville tutkimalla CHKDSK /V -komennon tulostamaa listaa. /V-valitsimella ohjattuna CHKDSK tulostaa kaikkien löytämiensä tiedostojen nimet niiden määreistä riippumatta. Valitsinta voi käyttää, vaikkei hakemistoa olisi tyhjennettykään, mutta tällöin näkyvien tiedostonimien vertailu CHKDSK:n tuottamaan listaan ja näkymättömien tiedostojen etsiminen saattaa käydä työlääksi.

Jos piilotuksen lisäksi tiedetään myös mahdollisen tiedoston nimi, on helpointa käyttää TYPEä. Edellisessä niksissä kerrottiin, että se tulostaa tiedoston ruudulle määreistä piittaamatta. Tätä voidaan käyttää apuna esimerkiksi selvittämään, onko mikrossa PC-DOSin vai MS-DOSin käyttöjärjestelmätiedostot: siirrytään C: aseman päähakemistoon ja annetaan komento TYPE IO.SYS. Mikäli TYPE tulostaa ruudulle roskaa, IO.SYS on olemassa ja levyllä on MS-DOS. Jos taas TYPE antaa virheilmoituksen eikä tulosta mitään, levyllä on PC-DOSista peräisin oleva IBMBIO.COM. Varman tuloksen saamiseksi pitää tutkia molemmat tapaukset, sillä jos levyllä on ennen ollut MS-DOS ja sen päälle on päivitetty PC-DOS tai päinvastoin, päähakemistosta saattavat löytyä molempien piilotetut tiedostot.

77

Ääkköset tiedostonimissä

PC:n käytön alkeiskursseilla neuvotaan yleensä välttämään skandinaavisia erikoismerkkejä eli niin sanottuja ääkkösiä tiedostonimissä. Suositus johtuu siitä, että DOSin ensimmäisessä versiossa sallittuja olivat

ainoastaan amerikkalaisten käyttämät kirjoitusmerkit, numerot sekä eräät erikoismerkit. Kuitenkin jo DOS 2.0:ssa tiedostonimiä laajennettiin siten, että Ä- ja Ö-kirjainten käyttö tuli mahdolliseksi.

Miksi niitä sitten vieläkin neuvotaan välttämään? Koska joillakin sovelluksilla on yhä vieläkin vaikeuksia niiden kanssa. Ohjelmat on usein tehty USAssa, eivätkä sikäläiset ohjelmoijat ole tulleet ajatelleiksi että tiedostonimissä olisi muita kirjaimia kuin A-Z.

Ääkköset ovat ohjelmoijan kannalta teknisesti hankalia. Perusaakkosissa muunnos pienestä kirjaimesta isoon tehdään vähentämällä kirjaimen ASCII-koodista arvo 32. Näin käyttäjän pienellä kirjoittama tiedostonimi saadaan helposti muunnettua vastaaville isoille kirjaimille ennen nimen lähettämistä DOSin tiedostopalveluille. Ääkkösten kohdalla merkkimuunnos ei kuitenkaan ole näin suoraviivaista, vaan jokaisella pienellä erikoismerkillä on oma vastaava iso merkkinsä eikä niiden välillä ole mitään sääntöä. Esimerkiksi pienen ä-kirjaimen koodi on 132 ja ison 142.

Nimekkäistä sovelluksista ainakin Lotuksen 1-2-3 DOS-versiolla on ongelmia ääkkösnimien kanssa. Taulukkolaskentaohjelman 2.x-versiot kyllä hyväksyvät tiedostonimet vaikka niissä olisi ääkkösiäkin, mutta nimet näkyvät ohjelmassa väärin. Saman ohjelman 3.x-versioissa ongelmaa ei ole.

Windowsissa tuki kansallisille erikoismerkeille on hoidettu paremmin, mutta nyt mukaan tulee uusi ongelma: Windowsin käyttämä ääkköskoodaus on erilainen kuin DOSin käyttämä. Jos tiedostonimi tallennettaisiin siinä muodossa kuin käyttäjä on sen antanut, kaikki nimessä olevat Ä- ja Ö-kirjaimet näkyisivät väärin DOSin DIR-käskyllä. Tämän vuoksi Windows-ohjelmien on itse muunnettava tiedostonimessä olevat ääkköset DOSin käyttämään muotoon ja takaisin Windows-muotoon kun tällainen tiedosto avataan.

Lähes kaikki Windows-ohjelmat osaavat tämän — eivät kuitenkaan kaikki. Jopa Windowsin omissa apuohjelmissa on eräitä, joille ääkkösten muuntaminen tuottaa ongelmia.

Summa summarum: ääkkösten käyttö saattaa joissakin harvoissa sovelluksissa tuottaa ongelmia, mutta sekä Windows että DOS eivät rajoita niiden käyttöä.

78

Nopein tapa kopioida tiedostoja

Kun kopioidaan suuri määrä tiedostoja asemasta tai hakemistosta toiseen, kannattaa käyttää mahdollisimman nopeata tapaa.

Perinteinen COPY *.* on hidas. Historiallisista syistä johtuen se lukee ja kirjoittaa tiedostoja 64 kilon palasissa, mikä jo yksistään riittää tekemään siitä hitaan. Lisää hitautta aiheutuu siitä, että kun tiedostoja luodaan yksi kerrallaan, käyttöjärjestelmältä kuluu enemmän aikaa kirjanpidon käsittelyyn ja uuden tilan etsimiseen tiedostoille.

XCOPY on huomattavasti nopeampi. Se lukee muistiin niin paljon tiedostoja kuin sinne mahtuu ja kirjoittaa ne sitten kerralla kohteeseen. Isoissa kopioinneissa ero saattaa olla kymmeniä prosentteja. Kannattaa siis opetella käyttämään XCOPYä COPYn sijaan. Lisääntyneen nopeuden ohella XCOPYllä on muitakin etuja, kuten kyky kopioida kokonaisia hakemistorakenteita.

Näppärä on myös XCOPYn kyky luoda kohdehakemisto tarvittaessa. Esimerkiksi komento

```
C:\>XCOPY A:*.* C:\UUSI\
```

luo UUSI-hakemiston kopioinnin alkaessa. Jos hakemisto on jo olemassa, tulostuu virheilmoitus ja viimeinen kenoviiva pitää jättää pois.

79

Välimuisti voi hidastaa kopiointia

Käytettiinpä mitä tapaa tahansa, välimuistiohjelman taustakirjoitus hidastaa kopiointiprosessia jopa 20-30 prosenttia jos kopioitavaa aineistoa on enemmän kuin mitä puskuriin kerralla mahtuu. Hidastus johtuu siitä, että kirjoituspuskurin tultua täyteen välimuisti joutuu purkamaan

aineistoa levylle tausta-ajossa, jolloin se hidastaa sekä uuden tiedon lukemista että vanhan kirjoittamista.

Taustakirjoitus nopeuttaa kopiointia vain silloin, kun kaikki kopioitava aineisto mahtuu kerralla puskuriin. Kun tiedostot on luettu, niiden kirjoitus alkaa tausta-ajona ja komentotulkki on heti valmis ottamaan vastaan uusia komentoja. Mutta jos uutta aineistoa joudutaan lukemaan kirjoituksen ollessa vielä kesken, molemmat kärsivät.

Jos siis kopioitavia tiedostoja on paljon, kiintolevyn taustakirjoitus kannattaa kytkeä pois päältä kopioinnin ajaksi. Lukuvälimuisti kannattaa pitää aina päällä, koska se nopeuttaa kirjanpitoalueiden käsittelyä. Smartdrv:n tapauksessa kannattaa siis antaa komennot seuraavasti:

```
SMARTDRV D
XCOPY C:\KUVAT D:\VARA
SMARTDRV D+
```

80

Vaihtoehtoiset asetustiedostot

Microsoft Wordin DOS-versio tallentaa MW.INI-tiedostoon viimeksi käsitellyn tiedoston nimen sekä monia ohjelman toimintaan liittyviä asetuksia, kuten näytölle valitut värit sekä kirjoitinmäärittelyt. Kun käytetään /L-käynnistysvalitsinta se muistaa myös, missä kohtaa tekstiä käyttäjä on viimeksi ollut ja osaa avata tiedoston seuraavalla käynnistyskerralla samasta kohtaa.

Tämä kaikki toimii hienosti silloin, kun käsiteltävät tekstit ovat kaikki samassa hakemistossa. Koska tiedostojeni määrä alkoi kasvaa, perustin ohjelmahakemiston alle kolme eri työhakemistoa: KIRJEET, JUTUT ja KIRJAT. Halusin, että jokaisessa hakemistossa olisi muistissa erikseen, mitä tiedostoa olin viimeksi muokannut ja mistä kohtaa.

Osoittautui, että tämä oli helppo hoitaa pienen komentojonon avulla. Jono KIRJEET.BAT kopioi ensin KIRJEET-hakemistossa olevan INI-tiedoston Wordin ohjelmahakemistoon ja käynnistää sen jälkeen Wordin. INI-tiedoston ansiosta se osaa asettaa oletushakemiston

kirjeiden luo ja avata siellä viimeksi käsitellyn kirjeen. Kun tekstinkäsittely loppuu, muuttunut INI-tiedosto kopioidaan takaisin hakemistoon. Jonosta KIRJE.BAT tuli siten seuraava:

```
COPY KIRJEET\MW.INI
WORD /L
COPY MW.INI KIRJEET
```

Loin vastaavat komentojonot myös kahdelle muulle hakemistolle ja homma oli valmis. Nyt saatoin muokata Wordin asetuksia vielä niin, että kirjeitä käsiteltäessä Wordin tausta oli sininen ja teksti valkoinen, kun taas kirjan kanssa työskentely tapahtui 43 tekstirivillä ja mustalla taustalla ja niin edelleen. Erillisten INI-tiedostojen ansiosta kaikki määritykset pysyivät muistissa.

Oikeaoppisempi tapa hoitaa sama asia olisi ollut siirtyä ensin haluttuun hakemistoon ja käynnistää Word niin, että se olisi löytänyt oikean INI-tiedoston oletushakemistosta. Tämä ei kuitenkaan toiminut, sillä koska tiedosto sisältää ohjelman käytön kannalta keskeistä tietoa, Word etsii sitä vain omasta hakemistostaan. Silloin ainoaksi vaihtoehdoksi jää huijata ohjelmaa niin, että INI-tiedostoa vaihdetaan lennossa.

Vaikka edellä kuvattu menettely onkin luotu Wordin DOS-versiota varten, sitä voi soveltaa kaikkien sellaisten ohjelmien kanssa jotka tallentavat toiminta-asetuksensa ulkoisiin tiedostoihin.

80 B

Samaa niksiä voi soveltaa myös silloin, kun peliohjelmalla on useita käyttäjiä, jotka haluavat seurata omaa edistymistään pelin parissa. Pelit on yleensä tehty niin, että ne tallentavat parhaat tulokset ("High score") ulkoiseen tiedostoon, esimerkiksi HIGH.DATiin. Eri pelaajien high score-tiedostot saadaan pidettyä erillään tekemällä edellä kuvatun kaltainen komentojono, esimerkiksi seuraavasti:

```
ANNE.BAT:
COPY ANNE.HIG HIGH.DAT
PELI
COPY HIGH.DAT ANNE.HIG
```

ja

```
VIRPI.BAT:
COPY VIRPI.HIG HIGH.DAT
PELI
COPY HIGH.DAT VIRPI.HIG
```

DAT-tiedostoja ei tarvitse kopioida omiin hakemistoihinsa kunhan pidetään huolta siitä, että eri käyttäjien tiedostot säilyvät levyllä muista poikkeavilla nimillä. Kun Virpi käynnistää pelin kirjoittamalla VIRPI, hän näkee oman tulostaulukkonsa. Vastaavasti Anne näkee vain omat tuloksensa.

81

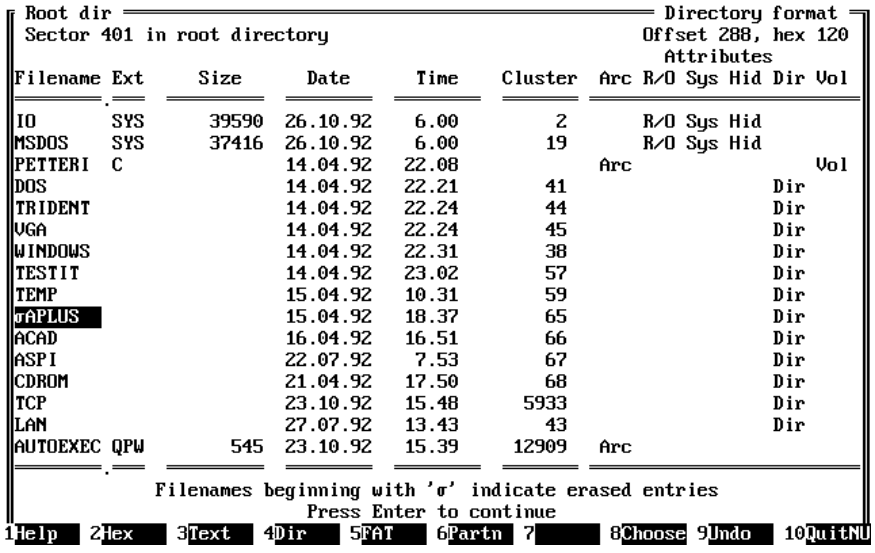
Hakemistohaaran poistaminen

Eräs DOSin komentotulkin kiusallisista rajoituksista on rekursiivisen poistokomennon puuttuminen. Monimutkaisen hakemistorakenteen poistaminen on työlästä, kun se pitää tehdä hakemisto kerrallaan: poistaminen pitää aloittaa puun alimmasta hakemistosta ja sieltä on edettävä järjestelmällisesti ylöspäin.

Jos käytössä on jokin levyeditori, kuten Norton tai PC Tools, kokonainen hakemistohaara saadaan poistettua kirjoittamalla ylimmän poistettavan hakemiston nimen alkuun koodi 229. Tästä koodista DOS kuvittelee, että hakemistonimi on poistettu eikä näytä sitä enää DIR-listauksessa.

Kirjanpito on vielä saatettava ajantasalle, jotta poistetuissa hakemistoissa sijainneiden tiedostojen käyttämä tila vapautuisi uuteen käyttöön. Koska CHKDSK näkee levyltä varatuiksi paljon alueita, jotka eivät näytä kuuluvan mihinkään tiedostoon, se raportoi ne orpoina varausyksikköinä. Ne palautetaan takaisin käyttöön käynnistämällä CHKDSK uudelleen /F-parametrilla ja vastaamalla kieltävästi kun ohjelma kysyy, luodaanko varausyksiköistä tiedostoja.

Hakemistonimen merkitseminen poistetuksi ja kirjanpidon "pakottaminen" kuntoon sen jälkeen on tehokas ja nopea, mutta epäelegantti tapa



Kokonainen hakemistorakenne alihakemistoineen saadaan poistettua kirjoittamalla hakemistonimen ensimmäisen kirjaimeksi sigma (koodinumero 229) ja pakottamalla sen jälkeen kirjanpito kuntoon CHKDSK/F:llä.

hoitaa hakemistorakenteen poistaminen. Jos mahdollista, poistaminen kannattaa mieluummin tehdä Nortonin ja PC Toolsin mukana tulevilla muilla apuohjelmilla.

82

Montako tiedostoa yhteen hakemistoon?

Kuten kohdassa 63 kävi ilmi, päähakemiston koko on kiinteä ja siihen mahtuu vain rajallinen määrä tiedostoja. Alihakemiston kohdalla on toisin. Yhteen hakemistoon voi sijoittaa vaikka miten monia tiedostoja. Teknistä ylärajaa ei ole, mutta mitä enemmän tiedostoja on, sitä hitaammaksi ja epäkäytännöllisemmäksi niiden käyttö muuttuu.

Varsinkin tekstinkäsittelyssä näkee usein tilanteita, että hakemistojärjestelmää tuntematon peruskäyttäjä tallentaa kaikki tekstinsä samaan hakemistoon ohjelman kanssa. Aikaa myöten tiedostoja saattaa kertyä satoja, jolloin oikean tiedostonimen löytäminen on työlästä ja päällekkäisyyksien välttäminen hankalaa.

Asiassa on myös tekninen puolensa. Kun käyttöjärjestelmä haluaa avata annetun nimisen tiedoston, se joutuu lukemaan koko hakemistolistauksen läpi alusta alkaen kunnes oikea tiedosto tulee vastaan. Hakemiseen kuluva aika kasvaa lineaarisesti tiedostojen kappalemäärän myötä.

Näiden ongelmien vuoksi kannattaa pyrkiä jakamaan tiedostot alihakemistoihin niin, että yhteen hakemistoon tulevien tiedostojen määrä ei ylittäisi 200:aa. Käytännön syistä rajan voisi vetää jo alemmaksikin. Ehdottomana ylärajana voisi pitää sitä, että kaikkien hakemiston tiedostonimien on mahduttava käytössä oleviin puskureihin — muuten tiedostolistan loppuosan lukeminen käy erittäin hitaaksi. Koska jokainen nimitieto vie 32 tavua, saadaan esimerkiksi 15:llä puskurilla raja-arvoksi 15*512/32 eli 240 tiedostonimeä.

Laskutoimitus koskee tilannetta, jossa ei käytetä välimuistia eikä ennakkopuskureita. Välimuisti poistaa listauksen käsittelyyn liittyvän hitauden, vaikka nimiä olisi paljonkin. Silti on parempi käyttää välimuistin nopeuttavaa vaikutusta tiedostojen käsittelyn eikä hakemistolistauksen nopeuttamiseen.

83

Kolme tapaa etsiä tiedostoja

Tiedoston etsimiseen nimen perusteella on olemassa kolme eri tapaa. DOS-versiosta riippuu, mitä niistä kannattaa käyttää.

Vanhoissa DOSeissa etsintä on kaikkein hitainta ja kömpelöintä. Lista levyn tiedostoista tulostetaan ensin aputiedostoon CHKDSK /V-komennolla ja sen jälkeen haluttu tiedostonimi etsitään siitä FIND-käskyn avulla. Korvausmerkkejä ei voi käyttää ja etsintä on muutenkin hidasta. Jos menetelmää joudutaan kuitenkin käyttämään, se kannattaa automatisoida komentojonoksi ETSI.BAT, joka on esitetty seuraavalla sivulla.

```
ECHO OFF
ECHO laadin tiedostoluetteloa
CHKDSK /V >C:\TEMP\FILEET.TXT
ECHO etsin tiedostoa %1:
FIND "%1" C:\TEMP\FILEET.TXT
ECHO Valmis!
DEL C:\TEMP\FILEET.TXT
```

DOS 3.3:stä alkaen etsintä käy huomattavasti helpommin ATTRIB-ohjelman avulla. /S-valitsinta käyttämällä ATTRIB käy läpi hakemistopuun oletushakemistosta alaspäin ja näyttää löytämänsä tiedostonimet. Sivutuotteena tulostuvat myös ehdot täyttäneiden tiedostojen määreet. Jos etsintä halutaan ulottaa koko levylle oletushakemistosta riippumatta, tiedostonimen eteen lisätään kenoviiva jolloin etsintä alkaa päähakemistosta alaspäin.

ETSI.BAT näyttää nyt seuraavalta:

```
ECHO OFF
ATTRIB \%1 /S
```

Jos esimerkiksi halutaan etsiä kaikki A:lla alkavat tiedostot, kirjoitetaan

```
ETSI A*.*
```

DOS 5.0:sta alkaen etsintä käy vielä nopeammin DIR-käskyn /S-valitsimella. Etsintä käynnistyy nopeammin, koska DIR on komentotulkin sisäinen käsky eikä sitä tarvitse ATTRIBin tapaan ladata levyltä. DIR /S ei myöskään tulosta tarpeettomia määretietoja ja kun etsintäkomentoon yhdistetään muita valitsimia, listaus saadaan halutussa muodossa.

Koska DOS 5.0:sta alkaen voidaan käyttää myös makroja, etsintätoiminto kannattaa määritellä makroksi. Esimerkiksi määritys

```
DOSKEY ETSI=DIR \$1 /S /B
```

etsii halutut tiedostot ja tulostaa niiden nimet lyhennetyssä (bare) muodossa.

83 B

Kaikissa edellisissä etsintätavoissa on yksi vika: ne kohdistuvat vain yhdelle levyasemalle kerrallaan. Jos mikrossa on useita asematunnuksia tai jos haku halutaan ulottaa myös verkon asemille, kannattaa luoda komentojono ETSI.BAT joka kohdistaa etsinnän halutuille asemille FOR-lausetta käyttäen:

```
ECHO OFF
FOR %%a IN (C: D: E: F: X: Y:) DO DIR %%a\%1 /S /P
```

Komento

```
ETSI A*.EXE
```

etsii kaikki A-alkuiset EXE-tiedostot annetuilta levyasemilta. Jos DOS-versio on vanhempi kuin 5.0, DIR-käskyn sijaan pitää kirjoittaa ATTRIB.

84

XCOPY+ATTRIB tiedostojen kopioinnissa

Tämä on jo vanha niksi, mutta otin sen mukaan kirjaan kun huomasin, että vain noin 10 prosenttia mikrotukihenkilöistä muisti sen asiaa kysyessäni.

Kun hakemistossa olevat monet tiedostot halutaan jakaa levykkeille, pelkkä COPY-käsky ei riitä työn tekemiseen. Se lopettaa kopioinnin heti, kun ensimmäinen levyke on tullut täyteen, eikä osaa jatkaa kopiointia seuraavalle levykkeelle.

BACKUP osaisi kyllä jakaa tiedostot eri levykkeille, mutta se vaatii parikseen RESTOREn eikä ole hyvä valinta silloin, kun ei tiedetä missä koneessa ja millä DOS-versiolla tiedostoja tullaan tarvitsemaan.

Ongelma ratkeaa näppärästi XCOPYn ja ATTRIBin yhteistyöllä kolmessa vaiheessa. Ensiksi asetetaan kaikkien kopioitavien tiedostojen arkistointimääre voimaan komennolla

```
ATTRIB +A *.*
```

Jos käytössä on DOS 5.0 tai uudempi, *.* merkintää ei tarvita.

Tämän jälkeen aloitetaan varsinainen kopiointi käyttämällä hyväksi XCOPYn /M-valitsinta:

```
XCOPY *.* A: /M
```

/M-valitsimella XCOPY ottaa kopiointiin vain ne tiedostot, joiden arkistointimääre on voimassa ja poistaa määreen heti kopioinnin jälkeen. Kun levy täyttyy ja XCOPY antaa siitä kertovan virheilmoituksen, asemaan vaihdetaan uusi levyke ja XCOPY-komento annetaan uudelleen. XCOPYä toistetaan levykettä vaihtamalla kunnes kaikki tiedostot on saatu kopioitua.

Koska XCOPY-komento säilyy kerrasta toiseen aivan samanlaisena, sen voi kutsua takaisin F3-näppäimen painalluksella. Valitettavasti tämä ei aina toimi, sillä kopioitavista tiedostoista riippuen XCOPY saattaa pyytää käyttöjärjestelmältä niin paljon muistia, että edellinen komentorivi hukkuu. DOS 5.0:sta alkaen kannattaa käyttää DOSKEYtä, joka suojaa muistin XCOPYllä ja varmistaa, että vanha komentorivi voidaan kutsua aina takaisin.

Yhtä tärkeätä kuin on niksin osaaminen on tietää, mitä rajoituksia siihen sisältyy:

- Kopiointi katkeaa ensimmäiseen tiedostoon, joka ei enää mahdu levykkeelle. Jos kopioitavana on paljon suhteellisen isoja tiedostoja, levykkeille saattaa jäädä paljon tyhjää tilaa.
- Yksikään kopioitavista tiedostoista ei saa olla isompi kuin mitä tyhjälle levykkeelle mahtuu. Jos tiedosto on liian iso, XCOPY ei koskaan pääse poistamaan sen määrettä kopioinnin merkiksi ja kopiointi jää ikuiseen silmukkaan.

85

Varausyksikön koon selvittäminen

Kun käyttöjärjestelmä pitää kirjaa levyllä olevista tiedostoista, se pyöristää tiedostojen koot aina ylöspäin. Käyttöjärjestelmä toimii kuten kaupan kassa: se ilmoittaa hinnat (= tiedostojen pituudet) tarkasti, mutta kun tulee maksun aika, tapahtuu pyöristäminen. Kaupan kassa pyöristää joskus myös alaspäin, mutta käyttöjärjestelmässä pyöristys tapahtuu aina ylöspäin. Toisin sanoen käyttäjä häviää aina.

Varausyksikön ja sen merkityksen tunteminen antaa loogisen selityksen monille DOSissa havaittaville ilmiöille. Siksi sen tunteminen on erittäin hyödyllistä. Mutta ensin on selvitettävä, mikä on omalla levyllä käytössä olevan varausyksikön suuruus. DOS 4.0:sta alkaen työ käy helposti, sillä CHKDSK kertoo asian

```
C:\NIKSIT>CHKDSK

Volume PETTERIN C  created 17.01.1992 20.24
Volume Serial Number is 16C7-98F4

  33419264 bytes total disk space
    155648 bytes in 9 hidden files
     81920 bytes in 29 directories
  31201280 bytes in 1059 user files
   1980416 bytes available on disk

      2048 bytes in each allocation unit
     16318 total allocation units on disk
       967 available allocation units on disk

    655360 total bytes memory
    586432 bytes free
```

Tätä vanhemmilla DOS-versioilla mittaus on tehtävä epäsuorasti. Aluksi kirjoitetaan muistiin, paljonko levyllä on vapaata tilaa. DIR-käsky näyttää tiedon nopeammin kuin CHKDSK, mutta tuloksen pitäisi olla sama:

```
          84 file(s)     1980416 bytes free
```

Tämän jälkeen luodaan levylle jokin tiedosto, joka on varmasti pienempi kuin yksi varausyksikkö. Koska pienin mahdollinen varausyksikkö on 512 tavua (eli yksi lohko), mikä tahansa tätä pienempi tiedosto ajaa asian. Tiedostoa ei tarvitse edes luoda, vaan sen voi kopioida:

```
C:\NIKSIT>COPY FREEMEG.COM TURHA.COM
```

Koska FREEMEG.COM-tiedoston pituus on vain 50 tavua, se sopii hyvin tarkoitukseen. Kopioinnin jälkeen katsotaan jälleen DIR-käskyllä vapaan tilan määrä:

```
      85 file(s)     1978368 bytes free
```

Kun luvut vähennetään toisistaan havaitaan, että 50 tavun tiedoston luominen levylle on vähentänyt vapaan tilan määrää

```
 1980416
-1978368
----------
    2048 tavulla.
```

Käyttöjärjestelmä on antanut syntyneelle tiedostolle yhden varausyksikön, jolloin vapaa tila on vähentynyt yhdellä varausyksiköllä eli kahdella kilotavulla.

Kun levylle perustetaan tiedostoja, käyttöjärjestelmä antaa niille tilaa aina kokonaisina varausyksiköinä. Esimerkkilevyllä jokainen alle kahden kilotavun mittainen tiedosto saa kaksi kilotavua. Jos tiedoston koko on vähän yli kaksi kiloa, sille annetaan kaksi varausyksikköä eli neljä kiloa ja niin edelleen. Näin tiedosto vie aina hieman enemmän tilaa levyltä kuin mitä sen pituudesta voisi päätellä.

Tämä johtaa eräisiin mielenkiintoisiin seurauksiin.

86

Varausyksikön hukkatilan selvittäminen

Jos levyn varausyksikkö on kaksi kilotavua ja levylle kopioidaan 12345 tavun mittainen tiedosto, lähin varausyksikköjen määrä johon se mahtuu on seitsemän, jolloin tilaa varataan 7 * 2048 = 14336. Kysymys kuuluukin nyt: mitä tapahtuu yli jäävälle tilalle?

Ja vastaus on yksinkertainen: se menee täysin hukkaan. Esimerkissä tiedoston lopun ja sille varattujen varausyksikköjen lopun väliin jää 14336-12345 = 1991 tavua, jota ei käytetä mihinkään. Hukkatila on väistämätön seuraus siitä, että kirjanpito ei ole täysin tarkkaa vaan perustuu pyöristykseen.

Miksei varausyksikköä sitten valita pienemmäksi? Mitä pienempi varausyksikkö, sitä vähemmän hukkatilaa tiedoston lopun ja sille varatun tilan lopun väliin pääsee jäämään. Valitettavasti tästä on kuitenkin omat haittansa: kirjanpidon koosta tulee sitä isompi, mitä useampia yksiköitä sen täytyy hallita. Tiedostot myös pirstoutuvat enemmän, jolloin niiden käsittely hidastuu.

Isosta varausyksiköstä on se hyöty, että kirjanpitoalueesta tulee pienempi ja tiedostot pirstoutuvat vähemmän. Haittana on hukkatilan lisääntyminen.

Varausyksikön koko on tyypillinen esimerkki "kun pyrstö irtoaa niin nokka tarttuu" -tilanteesta. Kumpikaan ääripää ei ole hyvä ja siksi käyttöjärjestelmä pyrkii tekemään asiassa kaikkia osapuolia tyydyttävän kompromissin. Käyttäjä itse ei pysty vaikuttamaan asiaan muuten kuin epäsuorasti tuntemalla periaatteen, jolla DOS valitsee käytettävän varausyksikön koon.

Äkkiseltään voisi luulla, että varausyksiköistä aiheutuva hukkatila on merkityksettömän pieni. Näin ei kuitenkaan ole. Jotta hukkaan menevän tilan määrä saataisiin laskettua tarkasti, pitäisi tutkia erikseen jokaisen kiintolevyllä olevan tiedoston pituus ja sille varatun tilan erotus. Koska tämä olisi varsin työlästä, voidaan tyytyä kohtuullisen hyvään arvioon. Osoittautuu, että hukkatilan määrää on helppo arvioida.

Kaikki perustuu yhden tiedoston kuluttamaan hukkatilaan. Koska emme tiedä, paljonko se on, laskemme hukkatilalle keskiarvon. Voidaan olettaa, että tiedostojen pituudet ovat jakautuneet tasaisesti ja tästä syystä jokainen tiedosto jättää käyttämättä keskimäärin puolet viimeisestä varausyksiköstä. Toinen puoli menee hukkaan.

Kun nyt tunnetaan hukkaan menevän tilan määrä tiedostoa kohti tarvitaan enää tiedostojen kappalemäärä. Kertomalla ne keskenään saadaan kohtuullisen hyvä arvio hukkatilan määrälle. Levyllä olevien tiedostojen määrä selviää CHKDSK-listauksesta:

```
Volume PETTERIN C  created 17.01.1992 20.24
Volume Serial Number is 16C7-98F4

  33419264 bytes total disk space
    155648 bytes in 9 hidden files
     81920 bytes in 29 directories
  31201280 bytes in 1059 user files
   1980416 bytes available on disk

      2048 bytes in each allocation unit
     16318 total allocation units on disk
       967 available allocation units on disk

    655360 total bytes memory
    586432 bytes free
```

Tarkkaan ottaen laskelmassa pitäisi huomioida myös piilotetut tiedostot ja alihakemistot, joiden määrä näkyy CHKDSK:n muilla riveillä, mutta koska kyse on vain tilastollisesta keskiarvosta, lopputuloskin on likiarvo.

Samaisen CHKDSK-listauksen loppuosa kertoi varausyksikön kooksi kaksi kilotavua. Siitä hukkaan menevä puolet on sopivasti yksi kilotavu, joten laskelmasta tulee helppo: 1059 tiedostoa tuhlaa kukin yhden kilon joten hukkaan menevä kokonaismäärä on 1059 kilotavua eli hieman yli megatavu.

Määrä saattaa kuulostaa paljolta, mutta kun sitä verrataan tiedostojen yhteenlaskettuun kokoon (29,7 megatavua) havaitaan sen olevan vain 3,48 % hyötytilasta. Määrä ei ole suuri kun ottaa huomioon, että sen ansiosta levyn tiedostot saadaan pidettyä järjestyksessä.

Kokonaan toiseksi tilanne muuttuu kun levyn kapasiteetti nousee. Teknisistä syistä kirjanpitoalueelle mahtuvien varausyksikköjen lukumäärä ei voi ylittää 65536:tta, joten isoilla levyillä jää ainoaksi mahdollisuudeksi kasvattaa varausyksikön kokoa ja tällöin hukkatilan määrä saattaa nousta hyvinkin suureksi.

Esimerkiksi yhteen C: tunnukseen jaettu 280-megainen kiintolevy tuottaa seuraavan CHKDSK-listauksen:

```
C:\>CHKDSK

Volume SCSI-ASEMA  created 28.06.1990 8.58
Volume Serial Number is 16CD-63AD

 293421056 bytes total disk space
   4308992 bytes in 6 hidden files
   1720320 bytes in 208 directories
 281124864 bytes in 5988 user files
     49152 bytes in bad sectors
   6217728 bytes available on disk

      8192 bytes in each allocation unit
     35818 total allocation units on disk
       759 available allocation units on disk

    655360 total bytes memory
    483856 bytes free
```

Levyllä on lähes 6000 tiedostoa ja varausyksikön koko on kahdeksan kilotavua. Jokainen 5988:sta tiedostosta tuhlaa keskimäärin puolet viimeisestä varausyksiköstä, joten hukkatilan kokonaismäärä on 5988*4096 eli 23,4 megatavua. Tältä levyltä menee siis hukkaan enemmän kuin mitä ensimmäisillä IBM AT-mikrojen 20 megatavun levyillä oli!

Itse asiassa todellisen hukkatilan määrä on vielä tätäkin keskimääräisarviota suurempi, sillä tiedostojen pituudet eivät ole jakautuneet tasan vaan enemmistö tiedostoista on pieniä eikä yllä edes varausyksikön puolikkaaseen. Siksi arvio hukkatilan määrästä on alakanttiin ja todellinen määrä on jonkin verran suurempi.

87

Hukkatilan minimointi

DOS valitsee varausyksikön koon itse eikä käyttäjä voi suoraan vaikuttaa siihen. Koko määräytyy loogisen levyaseman koon perusteella seuraavan taulukon mukaisesti:

Loogisen aseman koko	Varausyksikön koko
0-16 Mt	4 kt
16-128 Mt	2 kt
129-256 Mt	4 kt
257-512 Mt	8 kt
513-1024 Mt	16 kt

Taulukko pätee vain suoraan ohjattaville levyille. Mikäli levyn käyttö edellyttää CONFIG.SYSiin asennettavaa ohjaintiedostoa, sen käyttämä varausyksikkö voi poiketa taulukon arvoista. Myös pakatut levyt (Stacker, DoubleSpace ym.) ja vanhat DOS-versiot saattavat käyttää taulukosta poikkeavia arvoja. Esimerkiksi DOS 2.0 käyttää 20 megatavun levyllä 8 kilotavun varausyksikköä, koska FATin osoittimet ovat 12-bittisiä eikä niillä voida merkitä kuin reilut neljä tuhatta varausyksikköä. Jotta tällä määrällä saataisiin katettua koko levy, varausyksikön kokoa on pakko kasvattaa.

Jotta hukkaan menevän tilan määrä minimoituisi, kannattaa valita aseman koko väliltä 16-128 megatavua, koska tällöin varausyksikön koko on pienimmillään. Jos edellisen esimerkin 280 megatavun levy olisi jaettu vaikkapa kolmeen loogiseen asemaan, kukin vajaat sata megatavua, varausyksiköksi oli saatu kaksi kilotavua ja hukkatilaa olisi syntynyt enää 5,8 megatavua.

Vaikka tiedostot olisivat pysyneet samoina, levylle olisi jaon jälkeen näyttänyt ilmestyneen 17,6 megatavua lisää tilaa!

Toinen mahdollisuus hukkatilan vähentämiseen on käyttää jotakin kehittyneempää tiedostojärjestelmää. Valitettavasti DOSissa ei ole vaihtoehtoja, mutta Windows NT ja OS/2 pystyvät käyttämään myös kehittyneempiä NTFS- ja HPFS-tiedostojärjestelmiä. Niissä varausyksikön koko on yleensä 0,5 kilotavua, joten hukkaan menevän tilan määrä on selvästi pienempi. Toisaalta näissä tiedostojärjestelmissä kirjanpitoalueet ovat selvästi FATia isommat, joten tilansäästöä ei juurikaan pääse syntymään. Suunnittelijat ovat kenties ajatelleet, että isoilta levyiltä voikin varata hieman isomman alueen kirjanpitoa varten — tärkeintä on etteivät lukuisat pienet tiedostot kasvata hukkatilaa kohtuuttomasti.

88

Vapaa tila varausyksikköinä

Varausyksiköt antavat selityksen monille DOSin ilmiöille. Esimerkiksi DIR-listauksen lopussa näkyvä vapaan tilan määrä ei voi koskaan näyttää seuraavalta:

```
 5 file(s)      1000 bytes free
```

Eikä tällaiselta:

```
15 file(s)      10 bytes free
```

Vapaana olevan tilan määrä on aina jokin varausyksikköjen monikerta eli sen on aina oltava jaollinen 512:lla. Pienin lukema, joka DIR-listauksen lopussa voi esiintyä, on siis:

```
25 file(s)      512 bytes free
```

Tästä seuraava ylöspäin on 1024, sitten 1536 ja niin edelleen. Luku ei myöskään voi olla koskaan pariton, koska 512:n kaikki kerrannaiset ovat parillisia.

Jos jonain päivänä näet DIR-listauksen lopussa parittoman luvun niin... no, useimmiten käynti optikolla riittää.

89

Pienetkin tiedostot tuhlaavat tilaa

Varausyksiköstä johtuen tiedostot vievät enemmän tilaa kuin mitä pituuden perusteella voisi odottaa. Tällä on eniten merkitystä pienten tiedostojen kohdalla, koska niillä hukkaan menevän tilan määrä on suhteellisesti suurin. Asiaa tuntematon mikronkäyttäjä voi ajatella, että eihän noin pienet tiedostot mitään tilaa vie ja jättää ne tästä syystä tarpeettomasti levylle.

Esimerkiksi Windowsin käyttämät PIF-tiedostot ovat pituudeltaan 545 tavua, mutta niiden todellinen tilankulutus on paljon suurempi. Jos levyllä on 20 PIF-tiedostoa, niiden yhteenlaskettu pituus on 10900 tavua. Todellinen tilankulutus levyllä, jonka varausyksikkö on kaksi kiloa, on kuitenkin 40960 eli nelinkertainen. Ja jos varausyksikkö on neljä kiloa, tiedostot vievät jo kahdeksan kertaa sen tilan mitä pituudet yhteenlaskemalla voisi kuvitella.

DOS 5.0:sta lähtien DIR-listauksen loppuun tulostuu paitsi levyllä vapaana olevien tavujen määrä myös DIR-listauksessa näytettyjen tiedostojen yhteispituus. Tämä pituus saadaan laskemalla tiedostopituudet yhteen, joten se ei kerro mitään todellisesta tilankulutuksesta. Myös Windowsin File Manager ilmoittaa tiedostojen yhteispituuden, ei niiden viemää tilaa. Tämä saattaa tuottaa yllätyksiä kun tiedostot kopioidaan esimerkiksi levykkeelle.

90

Miksi tiedostot vievät liikaa tilaa?

Tyhjän HD-korpun kapasiteetti on 1457664 tavua. Tästä tiedosta on apua silloin kun tutkitaan, mahtuvatko kaikki halutut tiedostot kopioitavaksi levykkeelle. DOS 5.0:sta lähtien DIR-listauksen kertoma yhteispituus on kuitenkin laskettu suoraan tiedostoista, eikä se kerro paljonko tiedostot vievät tilaa levykkeelle kopioituna. Jos siis DIR-listauksen

loppu osoittaa lähes korpun kapasiteettia lähentelevää tavumäärää on lähes varmaa, etteivät kaikki tiedostot tule mahtumaan levykkeelle.

DIR-listauksen lopussa näkyvästä luvusta

```
61 file(s)     1444022 bytes
              31326592 bytes free
```

voisi päätellä, että tiedostot mahtuvat korpulle. Käytännössä ne eivät kuitenkaan mahdu, sillä jokainen tiedosto kuluttaa pituutensa lisäksi keskimäärin puolet yhdestä korpun varausyksiköstä eli 256 tavua. Esimerkin 61 tiedostoa kuluttavat siten 15616 ylimääräistä tavua mikä saa kokonaistilan nousemaan yli korpun kapasiteetin.

Sama ilmiö näkyy myös toisinpäin. Kun täydeltä levykkeeltä kopioidaan tiedostoja kiintolevylle, huomataan että kiintolevyn vapaa tila väheni enemmän kuin mitä levykkeen kapasiteetti oli. Isomman varausyksikön vuoksi tiedostot vievät kiintolevyllä enemmän tilaa kuin levykkeellä.

91

Vika-alueen eristäminen

Aikaa myöten levyille saattaa syntyä fyysisiä virheitä. Ne ilmenevät esimerkiksi tiedostoja kopioitaessa niin, että tiedoston lukeminen tai kirjoittaminen pysähtyy virheilmoitukseen

```
Read error reading drive C:
```

Ellei uusikaan kopiointiyritys auta, vika on vakava ja estää sillä kohtaa olevan tiedoston käsittelyn. Virheestä pääsee eroon vain perusalustamalla kiintolevy uudelleen, jolloin vika merkitään levyn sisäiseen kirjanpitoon tai alustamalla levy FORMATilla, jolloin vika merkitään bad sectorina DOSin omaan kirjanpitoon.

Koska molemmat menetelmät ovat vaivalloisia ja edellyttävät levyn tyhjentämistä, on hyvä osata niksi, jolla ongelmaa voidaan kiertää. Virheilmoituksen tuottanut tiedosto nimetään uudelleen vaikkapa

nimelle BAD.BAD ja suojataan System-määreellä. Se voidaan myös piilottaa näkyvistä Hidden-määrettä käyttäen. System-määre on tärkeä siksi, että ilman sitä levyn järjestelyohjelma saattaisi luulla tiedostoa tavalliseksi ja siirtää sen levyllä uuteen paikkaan. System-määre takaa, että tiedosto pysyy paikoillaan ja peittää levylle tulleen vika-alueen.

92

RECOVER voi tuhota levyn

RECOVER on apuohjelma, joka lukee vioittuneesta tiedostosta vielä kunnossa olevat osat ja luo niistä uuden tiedoston toiseen kohtaan levyä. Näin tiedostosta saadaan pelastettua ainakin osa. Tekstitiedoston tapauksessa osakin on parempi kuin ei mitään, mutta jos kyseessä on ohjelma- tai jokin muu binääritiedosto, se ei todennäköisesti toimi koska keskeltä puuttuu pala.

Jos virheellinen alue on tiedostossa VIKA.TXT, korjausyritys tehdään kirjoittamalla

```
RECOVER VIKA.TXT
```

Tämän jälkeen RECOVER ilmoittaa, montako tavua se on saanut pelastettua.

Väärin käytettynä RECOVER voi kuitenkin tuhota koko levyn. Tämän niksin varsinaisena aiheena onkin väärän käytön välttäminen.

Väärä käyttö liittyy tilanteeseen, jossa käyttäjä unohtaa antaa pelastettavan tiedoston nimen ja kirjoittaa sen paikalle levyaseman nimen. Tällöin RECOVER kuvittelee, että lukuvirhe on sattunut päähakemiston alueella ja yrittää luoda päähakemiston kokonaan uudelleen. Samalla koko olemassa oleva hakemistojärjestelmä tuhoutuu.

Komento

```
RECOVER B:
```

tuottaa ensin varoituksen

```
The entire drive will be reconstructed,
directory structures will be destroyed.
Are you sure (Y/N)?
```

ja jos käyttäjä vastaa myöntävästi, RECOVER pyytää vielä kuittauksen ennen kuin se aloittaa työnsä:

```
Press any key to begin recovery of the
file(s) on drive B:
```

Työn aikana RECOVER etsii kirjanpidosta tiedostoketjut ja tekee niistä keinotekoisia tiedostoja. Tiedostot saavat juoksevasti numeroidun nimen ja tarkentimena on aina REC. Jos tiedosto on ollut pirstoutunut, jokaisesta palasta syntyy oma tiedosto. Ne eivät takuulla toimi erillisinä.

Lopputuloksena levy näyttää seuraavalta:

```
 Volume in drive B has no label
 Volume Serial Number is 3D38-0EEC
 Directory of B:\

FILE0001 REC    659968 06.12.92    9.46
FILE0002 REC    275456 06.12.92    9.46
FILE0003 REC     65024 06.12.92    9.46
FILE0004 REC      1536 06.12.92    9.46
FILE0005 REC      3072 06.12.92    9.46
FILE0006 REC      2560 06.12.92    9.46
        6 file(s)    1007616 bytes
                      450048 bytes free
```

Tämän jälkeen käyttäjän on tutkittava jokainen tiedosto erikseen ja annettava sille oikea nimi.

Jos komento olisi annettu kiintolevyllä sen vaikutus olisi ollut vielä tuhoisampi, sillä kaikki alihakemistot olisi menetetty.

Miksi DOSiin on sitten jätetty näin vaarallinen apuohjelma? No, mikä tahansa ohjelma voi olla vaarallinen väärin käytettynä. Voihan esimerkiksi FORMATillakin tuhota koko kiintolevyn, jos ei tiedä mitä tekee. Vastaavasti RECOVER on vaarallinen vain silloin, kun käyttäjä ei tiedä miten sitä käytetään.

93

Tiedostojen kunnon tarkistaminen

Levyllä olevat fyysiset virheet tulevat esille vasta silloin, kun vian kohdalla olevaa tiedostoa yritetään käyttää esimerkiksi kopioimalla sitä paikasta toiseen tai käynnistämällä se. Varsinkin levykkeellä olisi hyödyllistä, jos voisi yhdellä komennolla tarkistaa että kaikki hakemiston tiedostot ovat kunnossa.

Tällainen komento on olemassa ja se on COPY tehostettuna /B-valitsimella. Komento

```
COPY SOFTA.EXE /B NUL /B
```

kopioi SOFTA.EXE-tiedoston tyhjälaitteeseen, mustaan aukkoon eli ei mihinkään. Vaikka COPY ei kirjoitakaan tiedostoa mihinkään, se lukee sen kuitenkin läpi ja lukiessa paljastuvat mahdolliset viat. Kun komentoon lisätään korvausmerkkejä, saadaan tarkistettua kaikki hakemiston tiedostot:

```
COPY *.* /B NUL /B
```

Tärkeätä: Joissakin lehdissä ja jopa alan kirjallisuudessa näkee komentoa käytettävän muodossa

```
COPY *.* NUL
```

Tämä ei kuitenkaan toimi oikein. Koska kopioinnin kohteena on laitenimi, DOS käsittelee tiedostoa tekstinä ja katkaisee kopioinnin heti ensimmäiseen vastaantulevaan Ctrl+Z -merkkiin (ASCII-koodi 26). Ohjelma-, kuva- ja muissa binääritiedostoissa on todennäköisesti useita Ctrl+Z-merkkejä, jolloin kopiointikäsky käsittelee vain tiedostojen alut eivätkä lopussa olevat mahdolliset virheet paljastu.

Valitettavasti ei ole mitään helppoa tapaa jolla kopioinnin saisi ulotettua kaikkiin levyn hakemistoihin yhdellä komennolla. XCOPYä ei voi käyttää, koska se ei suostu kopioimaan laitenimelle.

94

Siirtonopeuden mittaaminen

Tiedostojen kopiointia NUL-laitenimelle voi käyttää myös siirtonopeuden mittaamiseen. Kun tiedoston koko kiloissa jaetaan sen kopiointiin kuluneella ajalla, saadaan siirtonopeus kilotavuina sekunnissa. Varsinkin levykkeiden, optisten levyjen ja CD ROM -asemien nopeusmittaukset voivat antaa mielenkiintoisia tuloksia.

Kiintolevyllä mittaus voi antaa virheellisiä tuloksia, sillä DOS ei välttämättä pysty ottamaan vastaan tietoa niin nopeasti kuin mitä levyohjain pystyisi sitä lähettämään. Isoilla levyillä siirtonopeus voi myös vaihdella riippuen siitä, mistä kohtaa levyä mittaus tehdään. Uloimmilla urilla saatetaan käyttää suurempaa sektorimäärää kuin sisäurilla jolloin myös siirtonopeus on erilainen.

Kun siirtonopeutta mitataan pitää käyttää yhtä, mahdollisimman isoa tiedostoa. Jos tiedostoja on useita, kirjanpidosta aiheutuva viive kasvaa ja aika vääristyy niin, että siirtonopeus saa liian pienen arvon. CD ROMeilla ison tiedoston löytäminen mittaustarkoituksia varten ei ole mikään ongelma, sillä monilla levyillä on jopa yli 100 megatavun tietokantoja yms. tiedostoja. Seuraavat mittaukset on tehty kopioimalla reilun 22 megatavun pituista kuvatiedostoa COPY NUL /B-komennolla.

	aika	siirtonopeus
NEC -74	73 s	301 kt/s
Vanha Hitachi CD ROM	442 s	51 kt/s

Kuten mittaus osoittaa, NECin siirtonopeus ylittää niukasti valmistajan lupaaman 300 kilotavua sekunnissa. Vanha Hitachin asema omalla ohjaimellaan jää sen sijaan kolmasosaan siitä 150 kilotavusta sekunnissa, mitä CD ROM-standardi ja Windowsin multimediastandardi MPC edellyttäisivät. Näin siitä ei selvästikään ole enää nykyisten multimediasovellusten pyörittämiseen.

95

Smartdrv ja bad sectorit

Microsoft alkoi toimittaa parannettua Smartdrv-välimuistiohjelmaa ensi kertaa Windows 3.1:n mukana keväällä 1991. Aiempaan versioon nähden siinä oli monia parannuksia, kuten tuki irrotettaville levyille (ja levykkeille), taustakirjoitus ja toimintaparametrien säätömahdollisuus komentoriviltä. Alkuinnostuksen jälkeen eräät käyttäjät valittivat, että Smartdrv saattaa välillä hidastua selvästi ja näyttää jopa pysähtyvän.

Syy on todennäköisesti levyllä olevassa vika-alueessa eli bad sectorissa. Alue on merkitty oikeaoppisesti FATiin, mutta koska Smartdrv lukee levyä etukäteen, se saattaa yrittää lukea viallistakin aluetta. Mikäli näin käy, seuraukset riippuvat kiintolevystä. Useimmat yrittävät lukea viallista kohtaa useita kertoja ennen kuin luopuvat yrityksestä. Näiden uudelleenyritysten aikana kaikki muu toiminta on pysähdyksissä.

Ongelma voidaan estää niin, että Smartdrv:n käyttämä lohkokoko määritellään yhtä suureksi kuin käytössä olevan kiintolevyn varausyksikkö. Silloin Smartdrv lukee ja kirjoittaa levyä varausyksikkö kerrallaan ja välttää viallisiksi merkityt varausyksiköt.

Jos CHKDSK ilmoittaa varausyksikön koon esimerkiksi seuraavasti:

```
2048 bytes in each allocation unit
```

määritellään Smartdrv:n lohkokoko /E-valitsimella seuraavasti:

```
SMARTDRV 1024 512 C+ /E:2048
```

/E-parametrin laskeminen pienentää myös välimuistin kokoa. Jos lohkokoko pudotetaan neljäsosaan (oletusarvo 8192), pitää välimuistin kokomääritys kasvattaa vastaavasti nelinkertaiseksi jotta koko pysyisi samana.

Turhan päiten lohkokokoa ei kannata mennä pienentämään, koska se kasvattaa levyoperaatioiden määrää ja vähentää täten välimuistista saatavaa hyötyä.

96

Salaperäinen EA DATA-tiedosto

OS/2 luo jokaisen käyttämänsä levyn ja levykkeen päähakemistoon kummallisen EA DATA. SF -nimisen tiedoston. Sekä keskellä tiedoston nimeä että sen tarkentimen alussa on välilyönti, vaikka tiedostonimissä ei muuten välilyöntejä sallitakaan.

EA DATA sisältää tiedostojen laajennetut määreet (EA, Extended Attributes), kuten tiedoston tuottaneen ohjelman nimen sekä tiedostolle mahdollisesti tehdyn yksilöllisen kuvakkeen. Kiintolevyllä tiedoston koko saattaa kasvaa useisiin megatavuihin. Tiedosto syntyy myös jokaiselle levykkeelle, jolle kirjoitetaan OS/2:n alaisuudessa.

Jos OS/2:n käytöstä luovutaan, EA DATA kannattaa poistaa ainakin kiintolevyltä, jotta sen viemä tila vapautuisi uuteen käyttöön.

97

Virheellisten tiedostonimien poistaminen

Joskus hakemistossa saattaa näkyä tiedostoja, joiden poistaminen aiheuttaa pientä päänvaivaa. Esimerkiksi EA DATA. SF on piilotettu, mutta vaikka sen muuttaisi näkyväksi, ei tiedoston poistaminen käy suoraan DEL-käskyllä, sillä se ei hyväksy nimissä ja tarkentimissa olevia välilyöntejä. Edes Windowsin File Manager ei suostu poistamaan tiedostoja, joiden nimissä on välilyöntejä.

Tällaiset tiedostot on kuitenkin helppo poistaa käyttämällä DOSin korvausmerkkejä. Esimerkiksi EA DATA. SF voidaan poistaa komennolla

```
DEL EA*.*
```

tai jos hakemistossa on muita EA-alkuisia tiedostoja, korvaamalla välilyönnit kysymysmerkillä:

```
DEL EA?DATA.?SF
```

Erikoisen nimisiä tiedostoja voi sekä luoda että poistaa Basicistä käsin. Poistokomento on KILL ja se hyväksyy lähes mitä merkkejä tahansa kunhan ne kirjoitetaan lainausmerkkien väliin. OS/2:n käyttämä tiedosto saadaan poistettua kirjoittamalla Basicissä komento

```
KILL "EA DATA. SF"
```

Paluu Basic-tulkista komentotulkkiin tapahtuu komennolla SYSTEM. QBasicissä saman komennon voi antaa Immediate-ikkunassa, jolloin se toteutetaan välittömästi.

Poistaminen muuttuu vaikeammaksi, jos tiedostojärjestelmä on seonnut niin, että nimissä näkyy graafisia merkkejä tai muuta sotkua. Silloin ainoa mahdollisuus on tarttua tiedostonimissä vielä oleviin oikeisiin merkkeihin ja korvata muut kysymysmerkeillä. Johtuen tavasta, jolla DOS käsittelee korvausmerkkejä, tähti ei voi esiintyä korvattavan nimen alussa vaan alussa olevat virhemerkit on korvattava tarpeellisella määrällä kysymysmerkkejä.

98

Mistä tulevat piilotetut tiedostot?

CHKDSK raportoi levyllä olevien piilotettujen tiedostojen määrän ja niiden yhteispituuden esimerkiksi seuraavasti:

```
155648 bytes in 9 hidden files
```

Kun piilotiedostojen määrä lisääntyy, käyttäjä saattaa huolestua ja epäillä jopa virusta.

Tämä pelko on kuitenkin aiheeton. Viruksista vain muutama — kuten Cinderella — käyttää piilotiedostoja. Yleensä asialla ovat aivan tavalliset sovellukset, erilaiset apuohjelmat ja varsinkin DOS itse.

DOS 5.0:sta lähtien kaikki levyn piilotetut tiedostot on helppo tarkistaa komennolla

```
DIR /AH /S
```

Mitä eri tiedostot sitten ovat? Seuraavassa listassa on lueteltu eräitä yleisiä piilotettuja tiedostoja:

Tiedostonimi:	Tarkoitus:
IBMBIO.COM	käyttöjärjestelmän PC-DOS versio
IBMDOS.COM	
IO.SYS	käyttöjärjestelmän MS-DOS versio
MSDOS.SYS	
DESCRIPT.ION	4DOSin käyttämä pidennetty tiedostonimikenttä
MIRORSAV.FIL	DOSin ja PC Toolsin MIRROR
PCTRACKR.DEL	poistettujen tiedoston seurantaohjelma
STACKVOL.000	Stackerilla luotu näennäinen pakattu levyasema
DBLSPACE.BIN	DoubleSpace'lla luotu näennäinen pakattu levyasema
DBLSPACE.INI	edellisen toiminta-asetukset
386SPART.PAR	Windows 3.1 kiinteä virtuaalimuisti
EA DATA. SF	OS/2 laajennetut tiedostomääreet
WP DATA. SF	OS/2
SCANVAL.VAL	McAfeen SCANin laskema tarkistussumma kiintolevyn varatuista alueista

Jos levyllä on ennen ollut PC-DOS ja se päivitetään uudemmalla MS-DOSilla tai päinvastoin, C: aseman päähakemistossa saattavat näkyä kaikki neljä käyttöjärjestelmätiedostoa.

Varsinkin OS/2:ssa piilotiedostojen määrä saattaa kohota jopa satoihin, koska undelete-toiminto perustuu tiedostojen piilottamiseen. DEL-käskyn yhteydessä tiedostot piilotetaan, jolloin ne säilyvät edelleen levyllä mutta eivät enää näy hakemistossa.

99

Tiedoston ajan ja päiväyksen muuttaminen

Levyllä olevan tiedoston kellonaika ja päiväys saadaan vaihdettua komennolla

```
COPY /B SOFTA.COM+,,
```

Tiedosto saa sen päiväyksen ja kellonajan, joka on voimassa käskyn antohetkellä. Jos kyseessä on binääritiedosto, pitää käyttää /B -valitsinta koska muuten DOS lopettaa kopioinnin ensimmäiseen tiedostosta löytyvään Ctrl+Z:aan eli loppumerkkiin.

Kaikkien hakemistossa olevien tiedostojen kellonajan ja päiväyksen vaihtaminen tapahtuu kirjoittamalla

```
FOR %i IN (*.*) DO COPY /B %i +,,
```

Tästä komennosta on hyötyä ainakin silloin, kun toimitetaan ohjelmalevykkeitä asiakkaille. Asettamalla kellonajan sopivasti ennen komennon antamista saadaan DIR-listauksen näyttämän kellonajan paikalle ohjelman versionumero. Esimerkiksi kaikkien DOS 5.0:n ohjelmatiedostojen kellonaika on 5.00.

100

Käytä SHAREa

SHARE-apuohjelma on toimii muistinvaraisena ja kuluttaa 6,0 kiloa muistia. DOS 4.0:ssa kulutus on 100 tavua suurempi, sillä DOS 5.0:sta alkaen osa SHAREn toiminnoista on sisällytetty DOSin ytimeen. Kenties muistin säästämiseksi tai vain tiedon puutteesta johtuen monet PC-käyttäjät jättävät SHAREn lataamatta. Se on virhe.

SHARElla on kolme eri tarkoitusta:

- tukea FCBS-tiedostomekanismia yli 32 megatavun suuruisilla asemilla
- tehostaa DOSin kykyä havaita levykkeen vaihto
- mahdollistaa tiedostojen hallittu yhteiskäyttö

Ensimmäinen toimintamuoto koskee vain DOS 4.0:aa. Jos SHAREa ei ole ladattu, käyttöjärjestelmä antaa varoituksen "SHARE should be loaded for large media", mutta näyttää silti toimivan. Kuitenkin ne ohjelmat, jotka yhä käyttävät FCBS:iä, eivät pysty avaamaan yli 32 megatavun rajan meneviä tiedostoja.

DOS 5.0:sta lähtien tämä toiminta on lisätty DOSin ytimeen, joten SHAREa ei tästä syystä tarvita. Kaksi muuta kohtaa ovat kuitenkin riittävän hyvä syy ladata SHARE myös DOS 5.0:n jälkeen.

Toinen SHAREn tehtävistä on tarkkailla asemassa olevaa levykettä sen sarjanumeron perusteella. Jos sarjanumero puuttuu, tarkkailu tapahtuu levykkeen nimen perusteella. Kun levykettä vaihdetaan, SHARE pystyy kertomaan siitä sellaisillekin komennoille, jotka eivät itse tarkista levykettä (kuten komentotulkin COPY-käsky). Näitä ongelmia käsiteltiin kohdassa 69.

Viimeinen SHAREn tehtävistä on mahdollistaa hallittu tiedostojen yhteiskäyttö. Kun SHARE on muistissa, se pitää kirjaa avatuista tiedostoista. Jos kaksi eri ohjelmaa yrittää avata samaa tiedostoa yhtä aikaa sekä lukemista että kirjoittamista varten, SHARE antaa jälkimmäiselle pyytäjälle virheilmoituksen. Näin se suojaa tiedostoa. Monet tietokantaohjelmat eivät edes käynnisty, mikäli SHAREa ei ole ladattu.

Varsinkin Windowsissa on helppo tehdä se virhe, että sama työtiedosto avataan yhtä aikaa kahdella tai useammalla eri sovelluksella. Jos SHAREa ei ole ladattu, kaikki voivat käsitellä tiedostoa vapaasti. Viimeksi tehty tallennus kirjoittaa muiden päälle ja jää voimaan. Jos tiedosto on kuitenkin ollut pitkä, sen alku saattaa olla eri versiosta kuin loppu ja koko kirjanpito saattaa mennä tiedoston kohdalta sekaisin.

Näiden ongelmien välttämiseksi SHARE pitäisi aina ladata — varsinkin silloin kun käytetään jotakin moniajojärjestelmää.

100 B

Vaikka SHAREa suositellaankin erityisesti Windowsia käytettäessä on eräitä tilanteita, joissa se voi tuottaa ongelmia. Nämä tilanteet näkyvät usein erilaisina asennusongelmina, koska SHARE ei anna asennusohjelman avata jonkun toisen käytössä olevaa tiedostoa. Ongelmatilanteissa kannattaa siksi kokeilla, josko SHAREn poistaminen vaikuttaisi asiaan. Koska kerran muistiin ladattua SHAREa ei voi poistaa, kone pitää käynnistää uudelleen ja tällä kertaa ilman SHAREa.

100 C

Eräät tietokantaohjelmat saattavat vaatia tiedostojen yhteiskäyttöön varattujen lukkojen määrän kasvattamista, vaikka sekä tietokantaohjelma että sen tiedostot olisivat käyttäjän omassa koneessa. Esimerkiksi Microsoftin Access saattaa kieltäytyä tallentamasta tehtyä kyselyä (query) ja valittaa, että lukkoja on liian vähän.

Tämäkin ongelma ratkeaa SHAREn avulla. Komento

```
SHARE /L:100
```

varaa käyttöön sata lukkoa, minkä määrän pitäisi riittää.

101

Salaperäiset aputiedostot

"Oheisella levykkeellä on kaksi tiedostoa, joita en ymmärrä." Näin alkoi saatekirje, jonka sain eräältä huolestuneelta jyväskyläläiseltä lukijalta. "Tiedostot eivät näy suoraan DOSin alla, mutta listautuvat DIR|SORT-käskyllä. Tiedostoilla ei ole tarkenninta ja niiden koko on nolla. Ne kopioituvat välittömästi jokaiseen levykkeeseen, jota laitteistossa luetaan ja ne siirtyvät välittömästi levykkeiden mukana myös toisiin koneisiin.

Tiedostojen koko ei kasva, mutta nimi muuttuu joka listauksella. PC Tools ei niitä tavoita. McAfee-virusohjelma ei niitä myöskään

tunnistanut. En tiedä ovatko vaarallisia vai pelkästään harmillisia, mutta toivottavasti asia selviää."

Minun ei tarvinnut edes katsoa mukana tullutta levykettä koska tiesin heti, mistä oli kyse. Jälleen yksi mikronkäyttäjä, joka oli törmännyt DOSin luomiin tilapäisiin aputiedostoihin. Ja kuten lukemattomat mikroilijat ennen häntä, hänkin oli pelännyt niiden olevan merkkejä viruksista. Mistä siis oikein on kyse?

DOS perustaa levylle tilapäisiä aputiedostoja kahdessa tilanteessa: komentoputkissa ja vaihdettaessa ohjelmaa Shellin valikosta (DOS 5.0 alkaen). Varsinkin Shellin luomat aputiedostot saattavat olla suuria, sillä yhteen tiedostoon kirjoitetaan edellinen ajettu ohjelma eli käytännössä koko keskusmuistin sisältö. Aputiedostoilla ei ole tarkenninta ja niiden nimenä näkyy kahdeksan satunnaista numeroa tai kirjainta.

Yleisin tilanne, jossa aputiedostot putkahtavat näkyviin, on komentoputki

```
DIR | SORT
```

DOS ottaa DIRin tulostaman listauksen ja ohjaa sen SORTille. Toinen samanlainen tapaus on komento

```
CHKDSK /V | FIND ...
```

jota käytettiin ennen tiedostojen etsintään levyltä. Kun näitä komentoja suoritetaan, DOS tallentaa ensimmäisen komennon lähettämän syötteen aputiedostoksi levylle, ennen kuin syöte annetaan putken seuraavalle ohjelmalle. Nämä aputiedostot näkyvät tulostuvassa DIR-listauksessa:

```
 Directory of C:\NIKSIT\TESTI
 Volume in drive C is PETTERIN C
 Volume Serial Number is 16C7-98F4
.            <DIR>      05.12.92   19.51
..           <DIR>      05.12.92   19.51
BDDDBMEN               0 05.12.92   19.51
BDDDBMFN               0 05.12.92   19.51
CAPSLOCK COM          28 23.10.92   14.41
DOSVERS  COM           8 23.10.92   13.56
ISOKURSO COM          27 23.10.92   14.48
NUMLOCK  COM          28 23.10.92   14.41
SCRLOCK  COM          28 23.10.92   14.41
```

Aputiedostojen pituus näkyy nollana, koska tiedostot ovat olleet käsittelyhetkellä auki. Ja mikä merkillisintä: kun sama komento ajetaan uudelleen, tiedostojen nimet ovat vaihtuneet:

```
 Directory of C:\NIKSIT\TESTI
 Volume in drive C is PETTERIN C
 Volume Serial Number is 16C7-98F4
.            <DIR>      05.12.92   19.51
..           <DIR>      05.12.92   19.51
BDDFAOAG               0 05.12.92   19.53
BDDFAOBB               0 05.12.92   19.53
CAPSLOCK COM          28 23.10.92   14.41
DOSVERS  COM           8 23.10.92   13.56
ISOKURSO COM          27 23.10.92   14.48
NUMLOCK  COM          28 23.10.92   14.41
SCRLOCK  COM          28 23.10.92   14.41
```

Tavallinen DIR-käsky ilman putkikomentoa ei näytä niitä lainkaan. Ei ihme, että käyttäjä kuvittelee saaneensa viruksen.

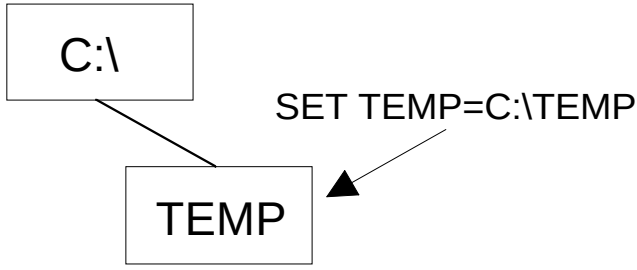
Toinen tilanne, jossa aputiedostot voivat tuottaa yllätyksiä, on komentoputken ajaminen kirjoitussuojatulle levylle. Esimerkiksi A: levykkeellä annettu komento

```
A:\>DIR | SORT
```

yrittää kirjoittaa aputiedoston oletuslevyasemaan eli kirjoitussuojatulle levykkeelle. Tämä ei tietenkään onnistu vaan tuottaa kirjoitussuojauksesta kertovan virheilmoituksen jolloin käyttäjä luulee, että koneessa on leviämistä yrittävä virus.

Jotta edellä kuvatuilta tilanteilta vältyttäisiin, versiosta 5.0 alkaen DOS tutkii ympäristömuuttujassa TEMP olevan hakemistonimen ja kirjoittaa aputiedostot sinne oletusasemasta riippumatta. Windows on käyttänyt samaa tekniikkaa alusta lähtien ja myös monet DOS-sovellukset noudattavat TEMP-muuttujan ohjeita. Eräät vanhat Windows-ohjelmat saattavat käyttää TMP-muuttujaa, mutta periaate on sama.

DOSin oma asennusohjelma asettaa TEMP-muuttujan osoittamaan DOS-hakemistoa, koska se on päähakemiston ohella ainoa hakemisto, jonka olemassaolosta voidaan olla varmoja. Windowsin asennusohjelma



Tilapäisiä aputiedostoja varten kannattaa perustaa oma hakemisto, johon tiedostot ohjataan TEMP-ympäristömuuttujan avulla.

perustaa WINDOWS-hakemiston alle TEMP-nimisen hakemiston samaa tarkoitusta varten.

Käyttäjän kannattaa kuitenkin perustaa itse TEMP-hakemisto ja ohjata aputiedostot sinne. Tämä tapahtuu lisäämällä AUTOEXEC.BATiin rivi

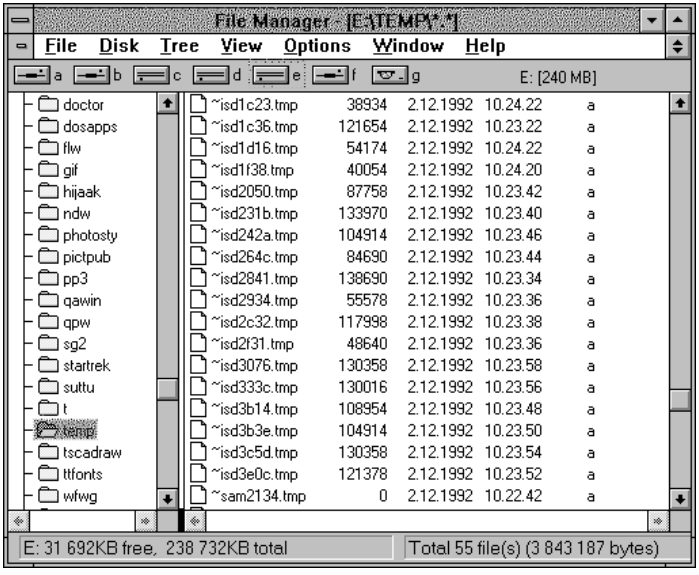
```
SET TEMP=C:\TEMP
```

Eräät vanhemmat ohjelmat saattavat käyttää TEMP-muuttujan sijaan TMP-muuttujaa. Siksi myös sen arvo kannattaa määritellä. Elegantimmin asia hoituu kirjoittamalla

```
SET TEMP=C:\TEMP
SET TMP=%TEMP%
```

Windowsin käyttämät aputiedostot on helppo erottaa niiden TMP-tarkentimesta ja tiedostonimen alussa olevasta aaltoviivasta. Tiedostojen pituus saattaa olla mitä tahansa muutaman kilon ja monen megatavun välillä. Aputiedostot näkee helposti esimerkiksi File Managerilla. Nollaa osoittava pituus kertoo, että tiedosto on parhaillaan auki ja sovelluksen käytössä.

Tiedostojen syntymisessä ei ole mitään yhtenäistä sääntöä eikä edes koneessa olevan muistin määrä vaikuta asiaan. Toiset sovellukset perustavat aputiedostoja ahkerasti, toiset eivät käytä niitä lainkaan. Kaikki



Windowsin ja Windows-sovellusten luomat aputiedostot on helppo tunnistaa TMP-tarkentimesta. Windowsin käytön aikana ne näkyvät esimerkiksi File Managerilla katsottaessa.

riippuu ohjelmointitavasta, jota ohjelmoija on käyttänyt. Windowsin omista ohjelmista aputiedostoja käyttää ainakin Print Manager, joka kirjoittaa jokaisen tulostusta odottavan työn ensin levylle aputiedostoon.

102

Aputiedostojen automaattinen poisto

DOS ja Windows sekä näissä toimivat sovellukset saattavat jättää levylle aputiedostoja, jotka varaavat turhaan kallisarvoista levytilaa. Yleensä aputiedostot poistuvat viimeistään silloin kun ne luonut ohjelma lopetetaan, mutta jos ohjelma jää jumiin tai katkeaa kesken käytön esimerkiksi sähkökatkon vuoksi, tiedostot jäävät poistamatta.

Koska aputiedostot perustetaan aina eri nimille, niillä on taipumus kasaantua. Esimerkiksi mikroissa, joissa on ongelmia Windowsin kanssa, saattaa olla kymmeniä aputiedostoja jotka syövät useita megoja turhaa tilaa.

Kuten edellisessä niksissä esitettiin, aputiedostot kannattaa ohjata ympäristömuuttujan avulla samaan hakemistoon, jotta ne olisi tarvittaessa helppo poistaa. Windowsin ja sen sovellusten luomat aputiedostot on helppo tunnistaa TMP-tarkentimesta. DOSin käyttämät aputiedostot ovat ilman tarkenninta, joten nekin on helppo tunnistaa. Sen sijaan DOS-sovellukset voivat luoda omia aputiedostojaan millä nimellä tahansa, joten niiden poistoa on hankala automatisoida.

Levylle mahdollisesti jääneet aputiedostot kannattaa poistaa aina kun mikro käynnistetään. Poiston voi tehdä huoletta, sillä on erittäin epätodennäköistä että tiedostoista löytyisi jotain käyttökelpoista. Poistokomennon voi lisätä esimerkiksi AUTOEXEC.BATin loppuun.

Poistokomento kannattaa kirjoittaa niin, että se itse tutkii TEMP-ympäristömuuttujan sisällön. Näin komentoa ei tarvitse muuttaa vaikka aputiedostot joskus ohjattaisiinkin toiseen hakemistoon. Poistokomennot kirjoitetaan seuraavasti:

```
DEL %TEMP%\*.TMP
DEL %TEMP%\~*.*
DEL %TEMP%\*.
```

Nämä rivit poistavat sekä kaikki TMP-loppuiset, aaltoviivalla alkavat että kaikki ilman tarkenninta olevat tiedostot hakemistosta, johon TEMP osoittaa. Ellei poistettavia tiedostoja ole, komennot tuottavat virheilmoituksen. Se voidaan estää tarkistamalla ennen poistoa, onko hakemistossa poistettavia tiedostoja:

```
IF EXIST %TEMP%\*.TMP DEL %TEMP%\*.TMP
```

Toinen vaihtoehto tyhjentää koko TEMP-hakemiston sisältö DEL *.* -komennolla, mutta tällöin tuhoutuvat kaikki hakemistossa olevat tiedostot.

103

Aja CHKDSK säännöllisesti

CHKDSK-apuohjelma vertaa levyn käytöstä kertovaa karttaa ja hakemistorakennetta toisiinsa. Jos kartta ja hakemistot vastaavat toisiaan, levyn kirjanpito on ainakin loogisesti kunnossa. Mikäli vastaavuudessa tai kartassa itsessään on virheitä, CHKDSK raportoi niistä käyttäjälle. /F-valitsinta käytettäessä CHKDSK yrittää jopa korjata osan näistä virheistä.

Valitettavasti käyttöjärjestelmässä ei ole mitään virheilmoitusta, joka suoraan kertoisi kirjanpitoon tulleista virheistä ("FAT corrupted" tai "FAT error"). Mahdolliset virheet näkyvät ainoastaan epäsuorasti ja joskus käyttäjän voi olla vaikeata päätellä, mistä bitti oikein puristaa.

Tyypillisiä oireita kirjanpitoon tulleista vioista ovat mm. seuraavat:

- Yritys käynnistää ohjelma tuottaa virheilmoituksen "Error in EXE-file" eikä mitään tapahdu.
- Sovellusohjelmat eivät pysty avaamaan työtiedostoja vaan antavat erilaisia virheilmoituksia.
- Ohjelma käynnistyy normaalisti, mutta jää sitten täysin jumiin.
- Tiedoston lopussa näkyy pätkä jostain toisesta tiedostosta.
- Mikro näyttää käynnistyvän normaalisti, mutta jää jumiin jo ennen kuin valmiusmerkki ehtii tulostua kuvaruudulle.
- Hakemistolistauksessa näkyvät tiedot, kuten päiväykset ja pituudet, sisältävät mahdottomia arvoja.
- Hakemistopuu tuottaa loppumattoman ketjun, jossa alihakemistossa näkyy aina uusia alihakemistoja — loputtomiin.

Kaikki nämä ovat merkkejä siitä, että kirjanpitoon on tullut vikaa. Vian selvittäminen onnistuu CHKDSK:llä. Valitettavasti CHKDSK on paljon parempi analysoimaan vikoja kuin korjaamaan niitä.

Jos siis ohjelmat tuntuvat käyttäytyvän omituisesti, aja aina ensin CHKDSK ja etsi vasta sen jälkeen virhettä muualta. Jos ohjelma esimerkiksi toimii väärin tai jää jumiin, on aivan turha lähteä asentamaan sitä uudelleen ennen kuin vika kirjanpidossa on korjattu.

Jotta kirjanpidon mahdolliset viat paljastuisivat ajoissa, ennen kuin ne ehtivät tuottaa harmia käyttäjälleen, kannattaa CHKDSK-ajo tehdä säännöllisesti, vähintään kerran viikossa. Mutta /F-valitsimen käytössä kannattaa olla varovainen.

104

Käytä /F-valitsinta vasta harkinnan jälkeen

Koska CHKDSK:n kyky korjata havaitsemiaan virheitä on huono, CHKDSK:tä *ei koskaan* pitäisi käynnistää ensimmäistä kertaa /F-valitsimella. Kun CHKDSK yrittää korjata virheitä, se toimii hyvin suoraviivaisesti ja tilanteesta riippuen korjausyritys saattaa tuottaa enemmän vahinkoa kuin hyötyä.

Juuri tästä syystä CHKDSK ei automaattisesti yritäkään korjata mitään vaan korjausta pitää erikseen pyytää /F-valitsimella. Ennen /F-valitsimen käyttöä kannattaa kokeilla kaikki muut tavat tilanteen korjaamiseen.

Jopa alan kirjallisuudessa näkee silloin tällöin suosituksen kirjoittaa CHKDSK /F valmiiksi komentojonoon, kuten AUTOEXEC.BATiin. Näin ei ikinä pitäisi tehdä, sillä jos kirjanpito on mennyt sekaisin, automaattisesti ajettu CHKDSK /F tuhoaa loputkin tiedostot pakottaessaan kirjanpidon jälleen loogisesti kuntoon. Paljon parempi lopputulos saadaan tutkimalla MIRRORin luomia aputiedostoja ja palauttamalla niistä toimivat osat kirjanpidosta.

Mitä virheitä CHKDSK:n sitten kannattaa antaa korjata? Tavalliset Lost Clusterit eli orvot varausyksiköt se hallitsee suvereenisti, mutta kaikki tätä vaativammat korjaukset pitäisi joko tehdä käsin tai antaa jonkin älykkäämmän ohjelman, kuten PC Toolsin DiskFixin tai Nortonin Disk Doctorin huoleksi. Vasta sitten, jos mitään parempaa ohjelmaa

ei ole saatavissa eivätkä käyttäjän/mikrotukihenkilön taidot riitä korjausten tekemiseen käsityönä, annetaan CHKDSK /F:n yrittää.

Tärkeätä: CHKDSK /F:ää ei saa koskaan käyttää Windowsin tai jonkin sovellusohjelman alta. CHKDSK kuvittelee, että ohjelmilla auki olevat tiedostot ovat kirjanpidon virheitä ja yrittäessään korjata niitä tuhoaa ne lopullisesti. DOS 5.0:sta lähtien /F-parametrin käyttö on tällaisessa tilanteessa estetty, mutta vanhemmissa versioissa se on yhä mahdollista.

Tässä jälleen yksi syy siihen, miksi /F-valitsinta ei koskaan pidä käyttää ensimmäisellä ajokerralla.

104 B

DOS 4.x ja 5.0-versioissa CHKDSK-ohjelmassa on virhe, joka saattaa johtaa kirjanpidon tuhoutumiseen kun CHKDSK yrittää korjata sitä. Myös UNDELETE-ohjelmassa on vastaavanlainen bugi. Ne ilmenevät vain silloin, kun loogisen levyaseman koko ylittää 65278 varausyksikköä.

Yksiköiden määrä selviää tutkimalla CHKDSK-listauksen kohtaa "total allocation units":

```
 2048 bytes in each allocation unit
65318 total allocation units on disk
```

Koska varausyksikköjen teoreettinen enimmäismäärä on 65528, vaaravyöhykkeessä ovat vain ne loogiset asematunnukset, joiden koko on välillä 127,5-128 Mt, 255-256 Mt, 510-512 Mt tai 1020-1024 Mt. Koska virhe ilmenee vain näissä harvoissa tapauksissa, vaara levyn tuhoutumisesta CHKDSK /F:n vuoksi on lähinnä teoreettinen. Kaikesta huolimatta Microsoft on korjannut vian DOS 5.0A-versiosta alkaen. Jos CHKDSK- ja UNDELETE-ohjelmatiedostojen päiväykset ovat syksyltä 1991 tai sen jälkeen, vika on korjattu.

Asia nousi yleiseen tietoisuuteen tammikuussa 1993, kun päivälehdet uutisoivat virheen näyttävästi otsikolla "Tietokoneiden DOS sisältää tuhoisan virheen". Uutisessa kerrottiin, että virheen vuoksi DOS saattaisi tuhota levyn tiedostoja viruksen tapaan. Virheen merkitykseen verrattuna uutisointi oli tehnyt kärpäsestä härkäsen.

105

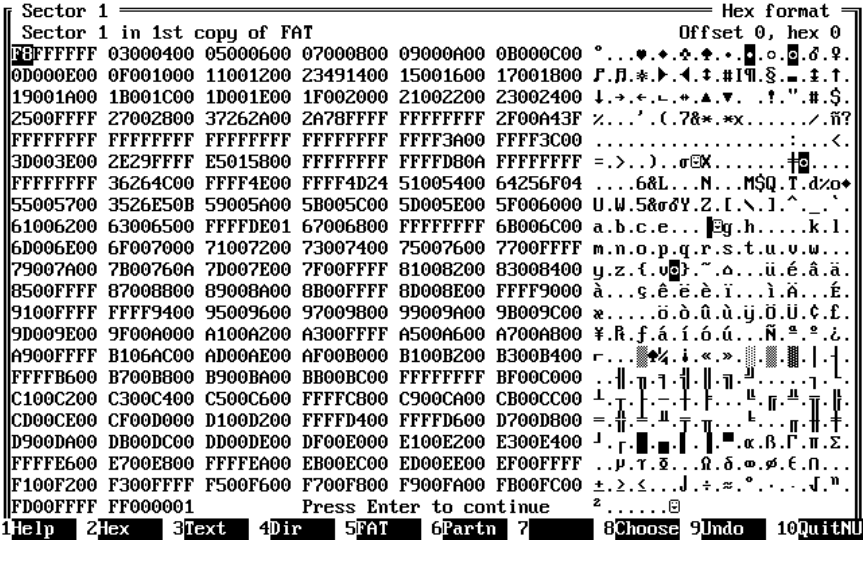
Probable non-DOS disk, continue?

Kun DOS tallentaa levylle tiedostoja, se pitää kirjaa käytetyistä ja vapaista alueista tilanvaraustaulukossa eli FATissa. Jotta tiedostoja voitaisiin lukea ja kirjoittaa, FATin pitää toimia virheettömästi.

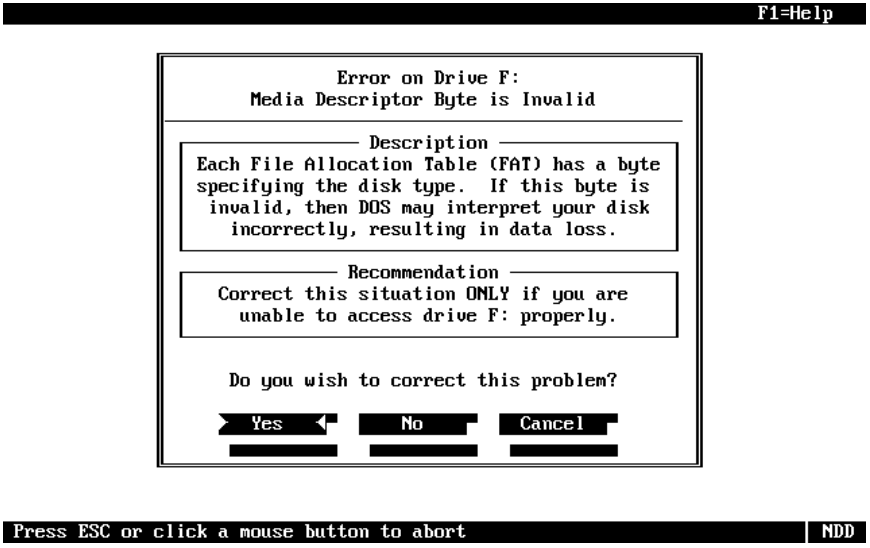
CHKDSK on ohjelma, joka tutkii FATin loogisen oikeellisuuden vertaamalla siitä löytyviä tietoja hakemistoihin ja tarkistamalla, että ne täsmäävät. Varmuuden vuoksi FATin alussa on tunnistetavu, jonka pitäisi kiintolevyllä olla aina F8 heksadesimaalisena eli 248 kymmenkantaisena. Jos CHKDSK huomaa, että tunnistetavu on viallinen, se antaa virheilmoituksen

```
Probable non-DOS disk, continue?
```

Koska tunnistetavu on virheellinen, CHKDSK arvelee että FAT saattaa olla kokonaan sekaisin ja kysyy, kannattaako sen edes yrittää FATin



Jos FATin alussa oleva tunnistetavu menee sekaisin, CHKDSK antaa varoituksen Probable non-DOS disk. Oikea arvo on kuvassa näkyvä F8.



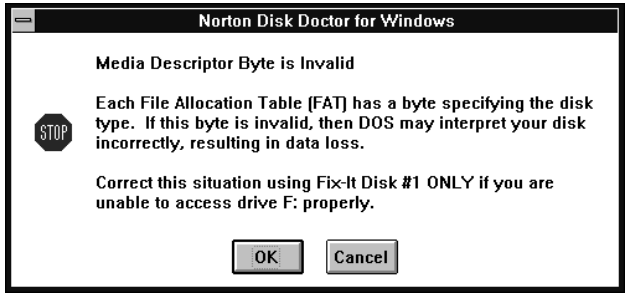
CHKDSK ei osaa korjata virheellistä tunnistetavua edes /F-valitsinta käytettäessä. Nortonin ja PC Toolsin korjausohjelmat selviävät työstä kuitenkin helposti.

analysoimista. Jos kysymykseen vastataan myöntävästi, CHKDSK jatkaa levyn kirjanpitoalueiden tutkimista normaaliin tapaan. Kielteinen vastaus keskeyttää tutkimuksen.

Tunnistetavun sekoaminen on usein merkki siitä, että koko FAT on sekaisin. Tällöin tiedostojen pelastaminen levyltä voi olla työlästä tai jopa mahdotonta. Kirjanpidon saa takaisin kuntoon vain alustamalla levyn uudelleen.

Onneksi virheilmoitus saattaa olla myös täysin vaaraton. Eräät ohjelmat saattavat nimittäin toimia virheellisesti niin, että levylle tehty kirjoitus ohjautuukin aivan levyn alkuun — FATin päälle — ja sotkee pelkän tunnistetavun. Koska muu osa FATista säilyy kunnossa, levyä voi käyttää normaalisti. Ainoa ongelma on CHKDSK:n antama kiusallinen virheilmoitus, josta ei pääse eroon edes CHKDSK /F-komennolla. CHKDSK ei osaa itse korjata tunnistetavua takaisin oikeaksi.

Ainoa tapa päästä eroon virheilmoituksesta on käyttää jotakin kehittyneempää korjausohjelmaa, kuten Nortonin Disk Doctoria tai PC Toolsin



Norton Desktopin levytohtori osaa tehdä oikean diagnoosin, mutta ei parantaa tautia. Windowsin käyttö pitää lopettaa ja korjaus tehdä sen jälkeen paketin mukana tulevalla DOS-ohjelmalla.

Diskfixiä. Ne korjaavat vian automaattisesti. Ellei tällaisia ohjelmia ole, korjauksen voi tehdä käsityönä millä tahansa ohjelmalla, joka pystyy lukemaan ja kirjoittamaan levyä suoraan. Korjausta tehtäessä on kuitenkin oltava erittäin varovainen, sillä muiden kuin ensimmäisen tavun muuttaminen FAT-alueella johtaa todennäköisesti tiedostojen vahingoittumiseen.

Hätätapauksessa vian korjaamista voi yrittää ilman kaupallisia apuohjelmiakin, pelkällä DOSin mukana tulevalla DEBUGilla. Tällöin on kuitenkin oltava hyvin varovainen, sillä väärään kohtaan tehty muutos saattaa tuhota koko kirjanpidon.

FAT-alueen alku (ensimmäinen looginen lohko) haetaan DEBUGin työtilaan komennolla

```
-L 100 2 1 1
```

Kun lohko on muistissa, se näytetään D-komennolla:

```
-d 100

1678:0100  F8 FF FF FF 04 00 26 00
1678:0110  09 00 0A 00 0B 00 0C 00
1678:0120  11 00 12 00 13 00 FF FF
1678:0130  19 00 1A 00 1B 00 1C 00
1678:0140  21 00 22 00 23 00 24 00
```

Kapean sivun vuoksi DEBUGin tulostuksesta on esitetty vain sen vasen puoli. Esimerkkilistauksen alussa näkyy lihavoituna oikea arvo F8 ja sitä seuraavat vakiotavut, joiden arvo on aina FF.

Jos arvo poikkeaa oikeasta, se kirjoitetaan komennolla

```
-e 100: F8
```

Tämän jälkeen FAT kirjoitetaan takaisin paikalleen komennolla

```
-w 100 2 1 1
```

Luku- ja kirjoituskomennon toinen numeroparametri tarkoittaa levyasemaa siten, että nolla vastaa A:ta, ykkönen B:tä ja kakkonen C:tä. Vian ollessa D: asemalla pitää numeron paikalle kirjoittaa kolmonen ja niin edelleen.

Yleensä virhe sattuu kiintolevyllä, jolloin tunnistetavun arvo on F8. Seuraavassa taulukossa on lueteltu muiden levytyyppien käyttämät arvot:

Levytyyppi	Tunniste-koodi
160 kt	FE
180 kt	FC
320 kt	FF
360 kt	FD
1,2 Mt ja 720 kt	F9
1,44 Mt	F0

106

Lost clusters eli orvot varausyksiköt

Tavallisin CHKDSK:n tulostamista ilmoituksista näyttää suunnilleen seuraavalta:

```
Errors found, F parameter not specified
Corrections will not be written to disk

5 lost allocation units found in 2 chains.
10240 bytes disk space would be freed
```

Lost cluster eli orpo varausyksikkö ei ole varsinainen virheilmoitus. Jos CHKDSK ei samalla raportoi muistakin virheistä, ei ole mitään syytä huoleen. Orvot varausyksiköt kuluttavat aivan turhaan osan (esimerkissä 10 kilotavua) levyn tilasta, mutta ne eivät estä levyn käyttöä.

Orpo varausyksikkö syntyy silloin, kun sovellus on ehtinyt avata tiedoston ja kirjoittaa sinne jotain, mutta ohjelman ajo on katkennut joko sähkökatkoon tai virheeseen ennen kuin tiedostoa on ehditty sulkea. Ja koska tiedostoa ei ole suljettu, sitä ei ole ehditty merkitä kunnolla kirjanpitoon. Vain ne alueet, joihin ohjelma on ehtinyt kirjoittaa, on merkitty varatuiksi levyn käytöstä kertovalle kartalle. Levyltä on siten varattu alueita, joita mikään tiedosto ei tunnusta omakseen.

Tällaisia alueita syntyy myös muiden kirjanpitoon tulevien virheiden seurauksena. Kun tiedoston sijainnista kertova ketju katkeaa tai viittaa virheelliseen kohtaan, CHKDSK tulkitsee ketjun loppuosan orvoiksi varausyksiköiksi.

Silloin kun kyse on pelkistä orvoista varausyksiköistä eikä muita virheitä ole, tilanne on helppo korjata. Kun CHKDSK käynnistetään uudelleen /F-valitsimella, se kysyy

```
Convert lost chains to files (Y/N)?
```

Myönteinen vastaus tekee orvoista varausyksiköistä väkisin tiedostoja. Koska CHKDSK ei voi tietää, minkä nimisiä tiedostojen on pitänyt olla ja mihin hakemistoon niitä on aikanaan yritetty perustaa, se keksii

tiedostoille itse nimet ja sijoittaa ne päähakemistoon. Jokaisesta ketjusta luodaan oma tiedosto ja ne nimetään FILE0000.CHK, FILE0001.CHK, FILE0002.CHK jne.

Vanhemmissa DOS-versioissa kysymys tulee joka kerta, mutta se otetaan huomioon vain jos /F-valitsinta on käytetty. Onneksi näin harhaanjohtava piirre on korjattu uudemmissa versioissa.

Yleensä kysymykseen kannattaa vastata kieltävästi, jolloin CHKDSK merkitsee kadonneet varausyksiköt heti vapaiksi, sillä tiedostopätkistä tuskin löytyy mitään hyödyllistä. Niitä tutkimalla voi kuitenkin saada käsityksen siitä ohjelmasta, mikä orvoksi jääneet varausyksiköt on synnyttänyt.

Jos orpoja varausyksiköitä syntyy säännöllisesti, vika on joko käytettävässä sovelluksessa tai sen käyttötavassa. Virta on kenties katkaistu kesken sovelluksen tai sitten käytetyt ohjelmat on tehty huonosti eivätkä ne sulje tiedostoja kunnolla. Myös levykkeen poistaminen asemasta merkkivalon vielä palaessa on varma tapa saada aikaan sekä orpoja varausyksiköitä että muita virheitä.

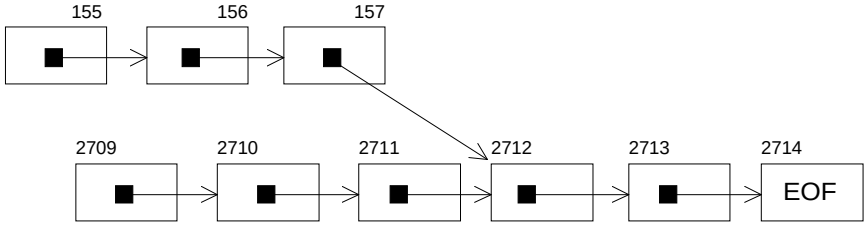
107

Ristiinlinkitetyt tiedostot

Orvot varausyksiköt ovat kiusallisia, mutta eivät mitenkään vaarallisia. Ristiin linkitetyistä tiedostoista kertova ilmoitus on sen sijaan aina merkki siitä, että ainakin yksi tiedosto on mennyt pahasti sekaisin — joskus kaksi tai useampiakin.

Virhe johtuu siitä, että tilankäytöstä kertovassa kartassa kahden tai useamman eri tiedoston linkkiketjut menevät päällekkäin. Kaikki ristiinlinkitetyt tiedostot väittävät, että kyseinen kohta levystä kuuluu niille. CHKDSK:n tulostamassa raportissa virhe näkyy seuraavasti:

```
C:\DOS\FORMAT.COM
   Is cross linked on allocation unit 2712
C:\KIRJA\PCNIKSIT.DOC
   Is cross linked on allocation unit 2712
```



Tiedostojen ristiinlinkitys tarkoittaa sitä, että kaksi eri tiedostoa väittää kuuluvansa samaan kohtaan levyä koska niiden sijainnista kertovat FAT-ketjut menevät päällekkäin. Virheen korjaaminen voi olla hankalaa eikä CHKDSK edes yritä sitä.

Koska tietty kohta levystä voi kuulua vain yhdelle tiedostolle kerrallaan, muut tiedostot ovat väärässä. Ongelma on vain löytää tiedostoista se, joka on oikeassa.

Koska ristiinlinkitysten korjaaminen on vaikeata ja edellyttäisi virheilmoituksen tuottavien tiedostojen sisällön tutkimista, CHKDSK ei edes yritä tehdä sitä. Vaikka ohjelman käynnistäisi /F-valitsimella, CHKDSK ei koskekaan levyllä oleviin ristiinlinkityksiin. Nortonin tai PC Toolsin apuohjelmat korjaavat kyllä ristiinlinkityksetkin, mutta paras lopputulos saavutetaan käsityönä seuraavasti:

Molemmat virheilmoituksessa esiintyvät tiedostot kopioidaan aluksi toiselle levylle ja poistetaan sen jälkeen alkuperäiseltä levyltä. Kopiointi toiseen alihakemistoon tai uudelle nimelle ei riitä, sillä ristiinlinkitysvirhe saattaa vaikuttaa koko asemaan. Tämän jälkeen tiedostojen kunto testataan avaamalla ne sovelluksella tai jos kyseessä on ohjelmatiedosto, kokeilemalla käynnistyykö ja toimiiko se normaalisti. Koska tiedostot ovat olleet päällekkäin on varmaa, että vain yksi niistä voi toimia oikein. Muut tiedostot ovat peruuttamattomasti pilalla ja ne tuhotaan tutkimuksen jälkeen.

Lopuksi pitää vielä ajaa CHKDSK /F, koska muuten ristiinlinkitetyiltä tiedostoilta jää levylle orpoja varausketjuja, jotka kuluttavat turhaan tilaa. Osaava käyttäjä voi Nortonin tai PC Toolsin avulla tutkia, olisiko orvoissa varausyksiköissä ristiin menneisiin tiedostoihin kuuluvia paloja, mutta tätä kannattaa yrittää vain jos tuntee täydellisesti

DOSin käyttämän tiedostojärjestelmän ja ristiinlinkitetyt tiedostot ovat olleet erittäin tärkeitä.

Edellä kuvattu korjausmenettely on suositeltava silloin, kun ristiinlinkitys koskee enintään muutamaa tiedostoa. Jos virheilmoituksia tulee useista tiedostosta tai jos ristiinlinkitysten lisäksi levy tuottaa muitakin virheilmoituksia on todennäköistä, että kirjanpito on mennyt kokonaan sekaisin. Tällöin tiedostojen kopiointi ei hyödytä mitään, koska linkkiketjut ovat virheellisiä myös tiedostojen sisällä. Ainoaksi vaihtoehdoksi jää yleensä levyn alustaminen uutta käyttöä varten.

108

Katoavien tiedostojen arvoitus

Aloittelevat PC:n käyttäjät ihmettelevät usein, miksi tiedostot katoavat levyltä itsekseen. Käyttäjä ei voi ymmärtää, miksei tiedostoa löydy levyltä, vaikka hän muistaa elävästi tallentaneensa tai kopioineensa sen edellisenä päivänä.

Katoavien tiedostojen arvoitukselle on monia selityksiä. Yhteistä niille kaikille on se, että käyttäjä ei tunne riittävän hyvin mikron tai sovelluksen toimintaa. Tiedostot eivät katoa levyltä itsekseen!

Yleisin syy tiedoston katoamiselle lienee se, että tiedosto tallentuu jostain syystä tavallisesta poikkeavaan hakemistoon. Käyttäjä joka ei tunne hakemistorakennetta, ei osaa etsiä tiedostoa oikeasta paikasta. Tällöin tiedoston etsimiseen kannattaa käyttää niksissä 83 esitettyjä tapoja.

Toinen syy voi olla ohjelman tulostamissa ilmoituksissa, joita käyttäjä ei osaa seurata. Levy voi esimerkiksi täyttyä tallennuksen aikana, jolloin ohjelma antaa siitä varoituksen. Käyttäjä, joka ei kiinnitä huomiota ohjelman antamiin ilmoituksiin, ei huomaa että mikään olisi vialla.

Myös DOSin komentotulkki saattaa tuottaa yllätyksiä. Komento

```
C:\UTIL> COPY NU.EXE \TOOLS
```

kopioi NU.EXE-tiedoston hakemistoon \TOOLS *jos hakemisto on olemassa*. Jos hakemistoa ei ole, COPY kuvittelee että käyttäjä haluaa kopioida NU.EXEn päähakemistoon ja antaa sen nimeksi TOOLS. Virheilmoitusta ei tule, koska komento on muodollisesti aivan oikea.

Vielä suuremman yllätyksen tuottaa komento

```
C:\UTIL>COPY *.* \TOOLS
```

sillä jos TOOLS-hakemistoa ei ole, COPY menettelee kerrassaan käsittämättömästi: se luo päähakemistoon yhden ison TOOLS-nimisen tiedoston, jossa on tiedostojen alut (ensimmäiseen Ctrl+Z:aan asti) liitettynä peräkkäin. Tällaisella tiedostolla ei tietenkään tee mitään.

Vanhoissa DOS-versioissa komentotulkki saattoi tuhota myös tiedostoja yrittäessään kopioida niitä itsensä päälle. Jos kohdemäärityksessä oli käytetty hakemistonimeä

```
C:\UTIL>COPY *.* \UTIL
```

komentotulkki ei huomannut asiaa vaan sotki tiedostot kirjoittaessaan niiden päälle.

Kaikilta näiltä ongelmilta välttyy käyttämällä XCOPYä tiedostojen kopiointiin. Tarvittaessa se kysyy, tarkoittaako käyttäjä tiedostonimeä vai hakemistoa ja estää näin vahingot.

109

Mikä määrää tiedostojen järjestyksen DIR-listassa?

Kukapa ei olisi joskus ihmetellyt, miksi uudet tiedostot joskus ilmestyvät keskelle DIR-listausta ja joskus taas sen loppuun. Mikä oikein määrää paikan, jonka levylle kopioitava tiedosto saa?

Kun hakemistossa olevia tiedostoja poistetaan, käyttöjärjestelmä tyhjentää tiedostojen DIR-listassa viemän paikan ja vapauttaa levytilan käyttöä kuvaavasta kartasta tiedostojen käyttämät alueet. DIR-käsky ei

tietenkään näytä listassa olevia tyhjiä rivejä vaan hyppää niiden yli. Näin käyttäjälle tulostuva lista näyttää aina yhtenäiseltä.

Kun samaan hakemistoon aikanaan tuodaan uusia tiedostoja, DOS antaa niille ensimmäisen hakemistosta löytyvän vapaan paikan. Jos hakemistosta on aikaisemmin poistettu tiedostoja, uudet tiedostonimet sijoitetaan niiltä vapautuneille paikoille. Jos taas yhtään tyhjää paikkaa ei ole, uudet tiedostonimet sijoitetaan listan loppuun.

DOS 5.0:sta lähtien tiedostolistaus saadaan tulostumaan aakkosjärjestyksessä käyttämällä valitsinta ja kirjoittamalla

```
DIR /ON
```

Jos komennon vaikutus halutaan pysyväksi, lisätään AUTOEXEC.BATiin rivi

```
SET DIRCMD=/ON
```

Tällöin komentotulkki lajittelee hakemistolistan tiedostot aakkosjärjestykseen jokaisella DIR-käskyllä. Komento ei kuitenkaan vaikuta itse tiedostolistaan ja siksi muut hakemistolistaa pyytävät ohjelmat näkevät tiedostot edelleen siinä järjestyksessä kuin ne sattuvat olemaan.

Pysyvää järjestystä varten tarvitaan Nortonin tai PC Toolsin apuohjelma, joka lajittelee tiedostolistan ja kirjoittaa sen takaisin hakemistoon.

110

Tiedostojen pysyvä poistaminen

Aloitteleva mikronkäyttäjä pelkää, että tiedostot katoavat levyltä heti, jos hän antaa yhdenkin väärän komennon. Kokenut käyttäjä tietää että asia on päinvastoin: tiedostoista on yllättävän vaikeata päästä eroon! Vielä senkin jälkeen kun tiedosto on poistettu DEL-käskyllä, se saattaa pulpahtaa näkyviin jopa puolivahingossa. Ja asiansa osaava tukihenkilö tai yksityisetsivä pystyy myös kaivamaan tiedoston esille

Miten siis menetellä, jos haluaa varmistaa ettei kukaan pysty nuuskimaan mikrolla tehdyn työn tuloksia? Voisi kuvitella, että on varminta

käsitellä tiedostoja koko ajan levykkeillä ja viedä levykkeet mukanaan työn jälkeen. Valitettavasti tämäkään ei aina riitä, sillä esimerkiksi monet tekstinkäsittelyohjelmat tekevät käytön aikana aputiedostoja ja välitallennuksia kiintolevylle vaikka tiedosto muuten olisi koko ajan levykkeellä. Näistä aputiedostoista ei ehkä pysty palauttamaan koko työtä, mutta ainakin niistä näkee mistä on ollut kyse. Ja mikäli toimitaan Windowsissa, virtuaalimuistin aputiedosto saattaa sisältää jäänteitä millä tahansa ohjelmalla käsitellystä aineistosta.

Levykkeet eivät siksi ole hyvä ratkaisu. Lisäksi niiden käytössä työtiedostojen muokkauksen aikana on omat vaaransa, kuten kohdassa 69 kävi ilmi.

Tiedostojen pyyhintään on kehitetty apuohjelmia, jotka tutkivat missä kohtaa levyä poistettava tiedosto on sijainnut ja kirjoittavat sen jälkeen samaan kohtaan levyä pelkkää nollaa. Eräät ohjelmat toistavat kirjoituksen varmuuden vuoksi useaan eri kertaan vaihtelevilla numeroarvoilla. Tämän jälkeen voidaan olla varmoja siitä, ettei tiedoston sisältöä pysty palauttamaan.

Vastaava toiminto on helppo ohjelmoida myös omatekoisena komentojonona. Kaikki perustuu kopiointikäskyyn, jolla kasvatetaan yhtä tiedostoa niin kauan että levy täyttyy. Kun levy tulee täyteen voidaan olla varmoja siitä, että käyttöjärjestelmä on ottanut käyttöön kaikki poistetuilta tiedostoilta vapautuneet alueet ja kirjoittanut niiden päälle.

Komentojono PYYHIALL.BAT näyttää seuraavalta:

```
ECHO OFF
ECHO Odota, pyyhin levyn vapaat alueet puhtaiksi...
:ALKU
COPY X+X XX
IF NOT EXIST XX GOTO VALMIS
DEL X
REN XX X
GOTO ALKU
:VALMIS
DEL X
ECHO Puhdasta tuli!
```

Pyyhkijänä käytetty tiedosto X voi sisältää mitä tahansa tekstiä eikä määrälläkään ole väliä. Jopa yksi ainoa merkki riittää. Jos haluaa hieman

näpäyttää levyn mahdollista tutkijaa, tiedostoon voi kirjoittaa vaikkapa tekstin:

```
Ähäkutti – levy on pyyhitty puhtaaksi!
```

Kopioinnin aikana se kirjoitetaan kaikille levyn vapaille alueille. Mikäli tiedosto ei ole tekstiä, DOSin COPY-käsky vaatii /B-valitsimen käyttöä jotta se suostuisi liittämään kaksi binääritiedostoa peräkkäin.

Komentojono täyttää levyn joten sen ajo kestää sitä kauemmin, mitä enemmän levyllä on vapaata tilaa. Koska pyyhkijänä käytettävän tiedoston pituus kaksinkertaistuu joka kierroksella, työhön kuluva aika riippuu pikemminkin levyn siirtonopeudesta kuin mikron prosessorin nopeudesta. Isokin levy tulee yleensä pyyhityksi muutamassa minuutissa.

Pyyhkimisen jälkeen tiedoston sisältö on mennyttä, mutta nimitiedot näkyvät yhä hakemistossa. Vanhat palautusohjelmat saattavat jopa väittää pystyvänsä palauttamaan tiedoston takaisin käyttöön. Jos näin tehdään havaitaan, että tiedosto kyllä palautuu mutta se sisältää vain X-tiedostossa ollutta merkkijonoa (kuten "Ähäkutti!").

Uudemmat palautusohjelmat osaavat käyttää DOSin poistonseurantaa (MIRROR) joka kertoo heti, että varausyksiköt on jo kierrätetty uuteen käyttöön. Silti esimerkiksi DOS 5.0:n mukana tulevassa UNDELETE-ohjelmassa on vielä bugeja, joiden vuoksi se välillä kuvittelee tiedoston palautuksen onnistuneen.

110 B

Käytettiinpä tiedoston pyyhkimiseen sitten edellä kuvattua komentojonoa tai jotakin kaupallista ohjelmaa, välimuistin kirjoituspuskuri pitää kytkeä pois käytöstä ennen kuin pyyhkiminen aloitetaan. Muuten voi käydä niin, että levylle tehty kirjoitus ei koskaan ehdi perille asti. Välimuistiohjelma huomaa, että samaan kohtaan levyä kirjoitetaan monta kertaa peräkkäin jolloin se fiksuna kuvittelee, että vain viimeinen kirjoituskerta on tarpeen.

Jos käytetään PYYHIALL.BAT-komentojonoa, sen alkuun kannattaa lisätä rivi

```
SMARTDRV C
```

ja loppuun vastaavasti

```
SMARTDRV C+
```

mikäli pyyhkimistä tehdään C: asemalla. Vielä käytännöllisempää olisi antaa pyyhittävän aseman tunnus komentorivillä, jolloin rivi

```
SMARTDRV %1
```

kytkisi kirjoituspuskurin pois käytöstä ja lopussa oleva rivi

```
SMARTDRV %1+
```

ottaisi sen takaisin käyttöön.

111

Tiedostojen palauttaminen

Kukapa ei olisi vahingossa poistanut levyltä vääriä tiedostoja? Vahinkoja sattuu kokeneillekin käyttäjille ja siksi on hyvä tietää, miten tiedostoja palautetaan ja miten niiden palautuvuutta voidaan edistää.

Kun DOS poistaa tiedoston, se muokkaa ainoastaan levyn kirjanpitoa. Tiedostonimi merkitään poistetuksi ja tiedoston aiemmin viemä tila merkitään vapautuneeksi. Tiedoston sisältöön ei kuitenkaan kosketa ja tämä mahdollistaa tiedoston palauttamisen.

Palauttamista varten on kehitetty monia apuohjelmia, joista Nortonin QU (Quick Unerase) oli ensimmäinen ja sai myöhemmin joukon jäljittelijöitä. DOS 5.0:sta lähtien UNDELETE-komento otettiin mukaan DOSiinkin ja myöhemmissä versioissa palautusta on vielä helpotettu.

Näillä työkaluilla tiedostojen palauttaminen on usein mahdollista, mutta ihmeisiin nekään eivät pysty. Jotta palauttaminen toimisi niin luotettavasti kuin suinkin, kannattaa muistaa seuraavat säännöt:

- Pidä levyllä mahdollisimman paljon vapaata tilaa. Se vähentää tiedostojen pirstoutumista, jolloin niiden palauttaminen helpottuu ja vähentää päällekirjoituksen vaaraa.

- Lisää AUTOEXEC.BATiin MIRROR /T-valitsimella, jolloin se jää muistinvaraiseksi ja alkaa seurata poistettujen tiedostojen nimiä ja niiden vapauttamia alueita.

Jos vahinko sattuu ja tiedostoja pitää palauttaa, muista seuraavat säännöt:

- Älä sulje mikroa, koska seuraavalla käynnistyskerralla DOS alkaa käyttää levyä alusta uudelleen ja saattaa kirjoittaa vapautuneen alueen päälle.
- Älä tallenna levylle uusia tiedostoja ennen kuin vanhat on palautettu.
- Jos joudut palauttamaan useita tiedostoja ilman MIRRORin apua, aloita palauttaminen pisimmistä tiedostoista, joihin tarvitaan eniten varausyksiköitä.
- Testaa kaikki palautuneet tiedostot ja varmista, että ne toimivat. Vaikka apuohjelma saisikin tiedostot palautettua, mikään ei takaa että kaikki oikeat varausyksiköt on saatu mukaan.

Mitä enemmän aikaa poistamisesta ehtii kulua, sitä pienemmäksi käyvät tiedoston palauttamismahdollisuudet, sillä DOS ehtii todennäköisesti ottaa vapautuneet alueet uuteen käyttöön.

112

Päällekirjoitetun tiedoston palauttaminen

Tiedostoa voi yrittää palauttaa silloinkin, kun on kopioinut vahingossa saman nimisen tiedoston toisen päälle. Vaikka uusi tiedosto DIR-listauksessa korvaakin vanhan, tiedostot eivät levyllä mene päällekkäin vaan DOS pyrkii sijoittamaan uuden tiedoston alkuperäisestä poikkeavaan paikkaan — jos vain levyllä on riittävästi tilaa.

Tiedoston vanhan version palauttaminen tässä tilanteessa on kuitenkin huomattavasti hankalampaa kuin edellä kerrotuissa tapauksissa, sillä

tiedoston uusi versio menee hakemistossa vanhan version päälle. Silloin menetetään vanhan tiedoston pituus- ja alkukohtatiedot, jotka ratkaisevasti helpottaisivat tiedoston palauttamista. Ilman näitä tietoja tiedoston alkukohta joudutaan etsimään levyltä käsityönä, mikä on erittäin hidasta ja hankalaa.

Mutta mahdotonta se ei ole. Ja jos on vahingossa kopioinut tärkeän tiedoston päälle toisen, saman nimisen tiedoston, kannattaa ainakin yrittää.

113

Kuinka iso tiedosto voi olla?

Jos kiintolevyllä vain riittää tilaa, niin miten ison tiedoston levylle voi luoda? Onko DOSissa jokin yläraja tiedoston pituudelle?

Raja toki on, mutta se on lähinnä teoreettinen. Ehdoton pituuden yläraja on hakemistolistauksessa näkyvä pituuskenttä, joka kertoo tiedoston pituuden niin käyttäjälle kuin käyttöjärjestelmällekin. Koska pituuskentälle on varattu neljä tavua, se riittää aina 4294967296 tavuun eli neljään gigatavuun asti.

Tätä aikaisemmin tulee todennäköisesti vastaan osion koon yläraja, joka DOS 5.0:sta alkaen on kaksi gigatavua. Ja vanhoissa, DOS 4.0:aa edeltävissä versioissa raja on ainoastaan 32 megatavua.

Jos vain levytilaa riittää, tiedoston koon kasvamista liian isoksi ei siis tarvitse pelätä.

114

TYPEn rajoitusten kiertäminen

Aloittelijoita muistutetaan siitä, että TYPE-käskyä tulee käyttää vain tekstitiedostojen tulostamiseen. Jos yritetään tulostaa binääritiedostoja, kuten kuvaa tai ohjelmaa, ruudulle tulostuu pelkkää roskaa ja kaiutin

piippailee ärsyttävästi. Tulostus loppuu ensimmäiseen tiedostosta löytyvään Ctrl+Z -koodiin eli ASCII-koodina 26:een.

Jos halutaan nähdä koko tiedosto loppumerkeistä huolimatta pitää käyttää /B-valitsinta ja tulostaa tiedosto komennolla

```
COPY tiedosto CON /B
```

Haittapuolena on, ettei tulostusta voi tällöin keskeyttää Ctrl+C:llä eikä Ctrl+Breakilla vaan se on kärsittävä piipityksistä huolimatta loppuun asti.

114 B

Toinen TYPEn rajoituksista on että sillä voi tulostaa vain yhden tiedoston kerrallaan. TYPE ei ymmärrä korvausmerkkejä. Tässäkin COPY tulee avuksi. Esimerkiksi komento

```
TYPE *.BAT
```

joka ei toimi, saadaan kierrettyä COPYn avulla seuraavasti:

```
COPY *.BAT CON
```

Elegantimmin sama asia hoituu kuitenkin pienen silmukan avulla:

```
FOR %I IN (*.BAT) DO TYPE %I
```

115

DOSKEYn epätavalliset makronimet

DOS 5.0:sta lähtien käyttöjärjestelmän mukana tuleva DOSKEY-apuohjelma lisää komentotulkkiin sekä makrot että komentohistorian eli mahdollisuuden nähdä vanhat komennot, kutsua ne takaisin ja muokata niitä.

Yleensä DOSKEYn makromääritys tehdään kirjoittamalla

```
DOSKEY D=DIR
```

jonka jälkeen pelkkä D riittää tuottamaan komennon DIR. Harva kuitenkaan huomaa, että kirjainten sijaan makron nimeksi voidaan määritellä myös välimerkkejä ja niiden yhdistelmiä. Tämä avaa uusia mahdollisuuksia makrojen nimeämiseen.

Esimerkiksi komento

```
DOSKEY \=CD \
```

määrittelee kenoviivan niin, että komento

```
C:\TEMP>\
```

siirtää käyttäjän suoraan päähakemistoon. Toinen, samantapainen määritys on

```
DOSKEY ..=CD ..
```

jolloin kaksi pistettä riittää siirtymiseen edellisen tason hakemistoon. Tällaisia oikoteitä on helppo keksiä itse lisää tarpeista riippuen.

Myös Ctrl-näppäimet voi määritellä uudelleen, mutta silloinkin niiden perään on painettava Enteriä ennen kuin komento suoritetaan. Esimerkiksi rivi

```
DOSKEY ^D=DIR
```

määrittelee, että Ctrl+D ja Enter näyttää DIR-listauksen. Hattumerkkiä ei tietenkään pidä kirjoittaa vaan se tulostuu ruudulle kun painetaan Ctrl+D.

Komentojonot

116

Tilapäinen PATH

Eräät sovellukset saattavat vaatia PATHiin useita hakemistoja. Koska PATHille varattu 127 merkin pituus loppuu helposti kesken, on syytä pyrkiä pitämään PATH niin lyhyenä kuin mahdollista ja ottaa käyttöön lisähakemistot vain tarvittaessa. Tämä tapahtuu helpoimmin ottamalla käyttöön tilapäinen PATH, joka on voimassa vain yhden sovelluksen käytön ajan.

Komentojonosta tulee hyvin yksinkertainen. Voimassa oleva PATH-asetus laitetaan muistiin toiseen ympäristömuuttujaan, haluttu polku määritellään ja sovellus käynnistetään. Kun sovellus loppuu, PATH otetaan muistista takaisin käyttöön ja muistikäytössä ollut ympäristömuuttuja poistetaan jotta se ei kuluta turhaan muistia.

Koko jonoksi tulee siis:

```
SET OLDPATH=%PATH%
PATH C:\DOS;C:\SOFTA;C:\SOFTA\ALI;C:\SOFTA\YLI
SOFTA
PATH %OLDPATH%
SET OLDPATH=
```

117

Hakemiston lisääminen PATHiin: ADDPATH

Yhden hakemiston lisääminen PATHin loppuun käy helposti komentojonolla ADDPATH.BAT, joka näyttää seuraavalta:

```
PATH=%PATH%;%1
```

Kun sitä kutsutaan esimerkiksi kirjoittamalla ADDPATH C:\JONO, se ottaa alkuperäisen PATHin, lisää sen perään puolipisteen ja tekstin C:\JONO sekä sijoittaa koko litanian takaisin PATH-asetuksen paikalle.

Joskus näkee, että sovelluksen asennusohjelma lisää AUTOEXEC.BATiin rivin

```
PATH=C:\PCTOOLS;%PATH%
```

Tällöin on kyse saman niksin käytöstä. Tapa on kuitenkin huono, sillä se näyttää rumalta eikä tarkista, mahtuuko PATHin perään enää lisää merkkejä. Jos PATHin pituus on jo lähellä ylärajaa, uusien hakemistojen lisääminen ei onnistu.

Hakemiston lisääminen PATHin perään on helppoa. Valitettavasti ei ole olemassa mitään yksinkertaista tapaa, jolla PATHista voitaisiin vastaavasti poistaa yksittäisiä hakemistoja. Pienellä komentojonolla ja editorin suosiollisella avustuksella voidaan kuitenkin muokata koko PATHia, kuten seuraava niksi osoittaa.

118

PATH-editori

Tämä komentojono EDITPATH.BAT kirjoittaa voimassa olevan PATHin levytiedostoon ja avaa sen käyttäjän valitsemalla editorilla. Esimerkissä on käytetty DOS 5.0:n EDITiä, mutta jokin pienempi ohjelma (kuten TED tai Qedit) sopisi vielä paremmin. Kun PATH on muokattu, EDITPATH ottaa sen käyttöön.

```
ECHO OFF
REM laitetaan ensin vanha polku muistiin
PATH >VANHA.BAT
PATH >UUSI.BAT
EDIT UUSI.BAT
CALL UUSI
ECHO Polkuasetus on nyt seuraava:
PATH
DEL UUSI.BAT
```

EDITPATH vaikuttaa luonnollisesti vain kulloinkin käytössä olevaan polkuun. Jos muutoksesta halutaan tehdä pysyvä, se pitää kirjoittaa AUTOEXEC.BATiin.

119

Koneen lukkiinnuttaminen

Jokainen mikroa käyttänyt tietää, että silloin tällöin kone saattaa jäädä jumiin niin pahasti, että ainoa tapa saada kone jälleen hallintaan on katkaista siitä sähkö.

Kevyt lukitus saadaan pistämällä komentojono ikuiseen silmukkaan:

```
ECHO OFF
BREAK ON
:LUUPPI
GOTO LUUPPI
```

Lukituksen voi keskeyttää painamalla joko Ctrl+C tai ainakin Ctrl+Break.

Perusteellisempi lukitus saadaan komennolla

```
CTTY COM1
```

Tämän käsky ohjaa komentotulkkia lukemaan syötettä näppäimistön sijaan COM1-portista. Ohjaus palautetaan takaisin näppäimistölle käskyllä CTTY CON, mutta koska komentotulkki ei enää lue näppäimistöä, komentoa ei voida antaa tavalliseen tapaan kirjoittamalla vaan sen pitäisi tulla COM1-linjalta.

Vapaaehtoisesta lukkiuttamiskäskystä voi olla hyötyä esimerkiksi silloin, jos tehdään virustarkistus komentojonosta käynnistämällä. Lukkiuttamalla kone viruksen löytyessä taataan, ettei käyttäjä pääse antamaan muita käskyjä ja näin levittämään virusta mahdollisesti vielä puhtaana oleviin tiedostoihin.

Esimerkiksi SCAN-ohjelma palauttaa komentojonolle nollan, jos kone on puhdas. Mikäli ERRORLEVEL palauttaa tätä suuremman

arvon, levyltä on joko löytynyt virus tai ohjelman ajossa on tapahtunut virhe, joka vaatii tukihenkilön kutsumista paikalle.

```
ECHO OFF
SCAN C:
IF ERRORLEVEL 1 GOTO OK
ECHO Koneessasi on joko virus tai etsintä on
ECHO keskeytynyt virheeseen.
ECHO Kone on nyt lukittu - soita tukihenkilölle
ECHO (puh. 1234) ja pyydä
ECHO lisäohjeita.
CTTY COM1
:OK
REM ok
```

CTTY-käskyn avulla saatava lukitus purkautuu vain kun kone käynnistetään uudelleen.

119 B

Samalla periaatteella voi myös suojata komentojonon niin, ettei käyttäjä pysty keskeyttämään sitä esimerkiksi tärkeän operaation keskellä. Ohjelmointitekniikassa tällaista kohtaa ohjelmakoodissa kutsutaan kriittiseksi alueeksi, joskin sen merkitys on silloin hieman erilainen ja liittyy moniajoon.

Esimerkki:

```
ECHO OFF
REM tähän normaalit komennot, sitten alkaa
REM kriittinen alue
CTTY COM1
REM Tästä eteenpäin komentojonon suoritusta ei voi
REM keskeyttää
REM näppäimistöltä (muuten kuin painamalla
REM Ctrl+Alt+Del)
COPY *.* D:\APU
REM Kopiointi tehty, palautetaan ohjaus takaisin
REM käyttäjälle
CTTY CON
```

120

Tehokas GOTO yhdistää komentojonot

Komentojonokielen hyppykäskystä saadaan tehokkaampi kun hypyn kohde otetaan parametrista tai ympäristömuuttujasta. Sitä ei tarvitse kirjoittaa komentojonoon valmiiksi, sillä komentotulkki pystyy laskemaan sen arvon vasta suoritushetkellä. Esimerkiksi rivi

```
GOTO %osoite%
```

saa suorituksen haarautumaan kohtaan, jonka nimi lukee ympäristömuuttujassa. Tätä ominaisuutta voi käyttää hyväksi monella tavalla.

Kohdassa 89 kävi ilmi, miten pienetkin tiedostot tuhlaavat kallisarvoista levytilaa. Jos levyllä on useita pieniä komentojonoja, ne kannattaa tilan säästämiseksi koota yhdeksi isoksi jonoksi. Ajettava kohta on sitten helppo etsiä GOTOn avulla.

Esimerkiksi jonot SKANNAA.BAT

```
ECHO OFF
SCAN C:
IF ERRORLEVEL 1 GOTO OK
ECHO Koneessasi on joko virus tai etsintä on
ECHO keskeytynyt virheeseen.
ECHO Kone on nyt lukittu - soita tukihenkilölle
ECHO (puh. 1234) ja pyydä lisäohjeita.
CTTY COM1
:OK
REM ok
```

ja ADDPATH.BAT

```
PATH=%PATH%;%1
```

saadaan yhdistettyä tiedostoksi KJONOT.BAT seuraavasti:

```
ECHO OFF
SHIFT
GOTO %0
```

```
REM --- Tästä alkaa jono SKANNAA.BAT ---
:SKANNAA
SCAN C:
IF ERRORLEVEL 1 GOTO OK
ECHO Koneessasi on joko virus tai etsintä on
ECHO keskeytynyt virheeseen.
ECHO Kone on nyt lukittu - soita tukihenkilölle
ECHO (puh. 1234) ja pyydä lisäohjeita.
CTTY COM1
:OK
GOTO ULOS
REM ---- Tästä alkaa jono ADDPATH.BAT ---
:ADDPATH
PATH=%PATH%;%1
:ULOS
```

Komentojonon alkuun lisätty SHIFT pudottaa komentoriviparametreja yhdellä alaspäin, jolloin ajettavan komentojonon nimestä tulee %0 aivan kuin se olisi käynnistetty omalla komennollaan. Jonolle annetut parametrit löytyvät sitten normaaliin tapaan muuttujista %1, %2 jne.

Kun halutaan ajaa SKANNAA.BAT, kirjoitetaan

```
KJONOT skannaa
```

ja kun polkuun halutaan lisätä uusi hakemisto, kirjoitetaan

```
KJONOT addpath D:\NORTON
```

Koska GOTO pitää isoja ja pieniä kirjaimia samanarvoisina, voi komentojonojen nimetkin kirjoittaa isoilla tai pienillä. Tekniikan ainoa rajoitus on siinä, että hyppyosoitteen oikeellisuutta ei mitenkään tutkita. Jos käyttäjä kirjoittaa virheellisen komentojononimen, KJONOT.BAT pysähtyy virheeseen "Label not found". Näin käy myös silloin, kun KJONOT käynnistetään ilman mitään komentojononimeä.

121

/S-valitsin DEL-käskyyn

DEL-käsky siivoaa levyltä tiedostoja luudan tavoin, mutta siinä on yksi paha rajoitus: poistaminen tapahtuu vain yhdestä hakemistosta kerrallaan. DEL-käskyssä ei ole /S-valitsinta, joka olisi tarpeen kiintolevyn kevätsiivouksessa ja vähän muulloinkin.

Onneksi /S-valitsinta on helppo emuloida komentojonon avulla. Jono perustuu pieneen kikkailuun, jossa poistettaviksi valittujen tiedostojen nimet kootaan ensin ATTRIBin avulla tiedostoksi ja sitä muokataan EDLINin avulla niin, että jokaisen rivin alkuun lisätään DEL-käsky. Lopuksi tiedosto ajetaan komentojonon tapaan, jolloin tiedostot poistuvat. Komentojono vaatii toimiakseen DOS 3.3:n tai uudemman, koska vanhemmissa versioissa ATTRIB-ohjelma ei tunne /S-valitsinta tai koko ATTRIB-ohjelmaa ei ole.

Komentojono aluksi varmistetaan, että kaikilla poistettavilla tiedostoilla on arkistointimääre voimassa. Koska myöhemmin lisättävä poistokomento perustuu juuri arkistointimääreen korvaamiseen DEL-käskyllä on tärkeätä, että määre on paikallaan. Ensimmäinen ATTRIB /S-komento tekee tämän.

Toinen ATTRIB-komento tuottaa listan kaikista tiedostonimistä ja kirjoittaa sen POISTOT.BAT-tiedostoksi. Samalla suodatetaan pois kaikki kirjoitussuojatut tiedostot (R), jotta niiden poistoyritykset eivät suotta tuottaisi virheilmoituksia. R-kirjaimen molemmin puolin on oltava välilyönnit, jotta se ei täsmäisi tiedostonimissä oleviin R-kirjaimiin. Koska komentojonoa tullaan todennäköisesti käyttämään turhien aputiedostojen, kuten BAKien poistamiseen, kirjoitussuojauksen tarkistaminen on todennäköisesti tarpeetonta.

Seuraavaksi muokataan tiedostolistaa niin, että jokaisen rivin alussa näkyvä arkistointimääreestä kertova A-kirjain korvataan poistokäskyllä DEL. Ohjeet tähän tulevat tiedostosta OHJE.KOM, joka näyttää seuraavalta:

```
1,9999R A Ctrl+Z DEL
E
```

Ensimmäinen rivi on kirjoitettava täsmälleen oikein: R välilyönti A välilyönti Ctrl+Z ja DEL. Loppumerkin Ctrl+Z lisääminen tiedostoon onnistuu helpoiten EDITin avulla ja käyttäen niksissä 53 kerrottua tapaa. Toisella rivillä on pelkkä lopetuskomento, joka saa EDLINin tallentamaan muokatun tiedoston ja lopettamaan muokkauksen.

Jos OHJE.KOM tulostetaan ruudulle TYPEllä nähdään vain tiedoston alkuosa, sillä TYPE lopettaa tulostuksen Ctrl+Z merkkiin luullen tiedoston loppuvan siihen.

Lopuksi käynnistetään POISTOT-komentojono, joka tekee varsinaisen tiedostojen poistamisen. Komentojono kokonaisuudessaan näyttää seuraavalta:

```
@ECHO OFF
IF %1x==x GOTO PUUTTUU
ECHO Kerään poistettavia tiedostoja
ATTRIB +a \%1 /S
ATTRIB \%1 /S | FIND /V " R " >poistot.bat
ECHO Muokkaan poistolistaa
EDLIN poistot.bat <ohje.kom >NUL
DEL poistot.bak
ECHO Poistan tiedostoja... odota
CALL poistot >NUL
DEL poistot.bat
GOTO LOPPU
:PUUTTUU
ECHO kirjoita DELS *.bak, jos haluat poistaa
ECHO kaikki BAK-tiedostot eri hakemistoista
:LOPPU
```

Poistokomentoa käytetään esimerkiksi kirjoittamalla

```
C:\>DELS *.BAK
```

jolloin se poistaa BAK-tiedostot kaikista C: aseman hakemistoista. Varmuuden vuoksi alkutarkistuksiin kannattaa lisätä IF %1x==*.*x GOTO eikay, koska jos käyttäjä kirjoittaa vahingossa DELS *.*, kaikkien hakemistojen kaikki tiedostot poistuvat ja vahinko voi olla suuri.

122

Hitaat REM-lauseet

Jos komentojonossa on paljon kommenttirivejä, komentotulkilta kuluu turhaan aikaa niiden ohittamiseen. Toisaalta kommentteja kannattaisi käyttää runsaasti, sillä ne helpottavat komentojonon ylläpitämistä myöhemmin.

Hitaiden REM-lauseiden ongelman voi kiertää niin, että kommentit kirjoitetaan tiedostoon normaalin tekstin tavoin ja ennen kommenttien alkua sijoitetaan hyppykäsky, joka ohittaa ne, esimerkiksi seuraavasti:

```
ECHO OFF
GOTO OHITUS
Tämän komentojonon on kirjoittanut BATMAN,
joka on komentojonojen arvostettu asiantuntija jo
60-luvulta lähtien.

Tämä komentojono kopioi kaikki muuttuneet
tiedostot C: asemalta D:lle ja nollaa kaikkien
kopioimiensa tiedostojen arkistointimääreet.

Huom: Komentojono ei mitenkään tarkista, että
D:llä on riittävästi tilaa. Huolehdi siis itse
tarkistuksesta ennen komentojonon käynnistämistä.

:OHITUS
REM Varsinainen komentojono jatkuu tästä
```

123

Kaksoispiste korvaa REMin

Silloin, kun erityistä kommenttilausetta halutaan jostain syystä käyttää, voidaan REMin sijaan kirjoittaa tavallinen kaksoispiste. Paitsi että se säästää kirjoitusvaivaa, se tekee myös kommenttien lukemisen helpommaksi.

Kun rivin alussa on kaksoispiste, DOSin komentotulkki pitää sitä hyppykäskyn osoitteena eikä yritäkään tulkita rivillä olevia komentoja. Edellisen esimerkin kommentit voitaisiin siten kirjoittaa myös muodossa

```
ECHO OFF
:Tämän komentojonon on kirjoittanut BATMAN,
:joka on komentojonojen tunnustettu asiantuntija jo
:60-luvulta lähtien.

:Tämä komentojono kopioi kaikki muuttuneet
:tiedostot C: asemalta D:lle ja nollaa kaikkien
:kopioimiensa tiedostojen arkistointimääreet.

:Huom: Komentojono ei mitenkään tarkista, että
:D:llä on riittävästi tilaa. Huolehdi siis itse
:tarkistuksesta ennen komentojonon käynnistämistä.

REM Varsinainen komentojono jatkuu tästä
```

124

Käynnistyslokin pitäminen

Jos halutaan seurata mikron käyttöä, luodaan LOKIIN.BAT-komentojono, jota kutsutaan AUTOEXEC.BATissa. Tämä jono kirjoittaa käynnistyksen kellonajan ja päiväyksen lokitiedostoon. Kellonaika ja päiväys saadaan helposti TIME- ja DATE-komennoilla. Jotta DOS ei jäisi odottamaan vastauksia, kuittaus luetaan CR.CR-tiedostosta. Tämä tiedosto sisältää pelkästään rivinvaihdon.

Koska TIME- ja DATE-komennot tulostavat voimassa olevan kellonajan ja päiväyksen ensimmäisellä rivillään, tämä rivi suodatetaan FIND "Current" -komennolla ja ohjataan tiedostoon. Ohjausmerkkinä pitää käyttää kahta suurempi kuin -merkkiä, koska uudet tekstit lisätään vanhojen perään.

```
ECHO ** KÄYNNISTYS >>loki.dat
TIME | FIND "Current" <CR.CR  >>loki.dat
DATE | FIND "Current"  <CR.CR >>loki.dat
```

Kun komentojonoa kutsutaan, se kirjoittaa LOKI.DAT-tiedostoon rivit

```
** KÄYNNISTYS
Current time is  21.30.19,13
Current date is Fri 24.01.1992
```

Mikäli käytössä on suomennettu DOS-versio, "Current"-sanat pitää muuntaa suomenkielisten TIME- ja DATE-komentojen käyttämään muotoon.

125

Kuka oli viimeksi koneellasi?

Onko joku käyttänyt mikroasi sillä välin kun olit poissa? Isoissa yrityksissä kysymys saattaa olla hyvinkin aiheellinen, sillä illalla, viikonloppuna tai matkoilla ollessasi kuka tahansa saattaa käydä koneellasi ja kopioida sieltä tiedostoja. Asialla voi olla utelias työtoveri, mutta myös vahingontekijä tai suoranainen teollisuusvakooja.

Tunnetaan jopa tapaus, jossa siivooja levitti yrityksen mikroihin viruksen. Siivotessaan toimistoa viikonloppuna hän otti poikansa mukaan työpaikalleen. Koska yrityksessä oli tehokkaita mikroja, poika käytti tilaisuutta hyväkseen ja pelasi koneilla mukanaan tuomia pelejä äidin tehdessä työtään. Pojan pelatessa hänen levykkeillään majaillut tietokonevirus saastutti yrityksen mikrot.

Edellisen esimerkin komentojonoa voi soveltaa tähän ongelmaan niin, että AUTOEXEC.BATissa tulostetaan aina edellisen käynnistyskerran kellonaika.

Se kirjoitetaan muistiin AUTOEXEC.BATin riveillä

```
ECHO Edellinen käynnistys: >viime.dat
TIME | FIND "Current" <CR.CR  >>viime.dat
DATE | FIND "Current"  <CR.CR >>viime.dat
```

Tätä ennen pitää tietenkin tulostaa vanha, edellisellä käynnistyksellä luotu tiedosto:

```
TYPE VIIME.DAT
```

Se tulostaa ruudulle ajan, jolloin mikro on viimeksi käynnistetty. Aikaa seuraamalla on helppo havaita, onko mikroa käytetty esimerkiksi viikonlopun aikana.

126

Hakemiston tyhjentäminen

Hakemistopuun siivoukseen tarvitaan suuri määrä DEL- ja RD-komentoja. Työtä helpottaa komentojonon POISTA.BAT, joka ensin tyhjentää parametrina saamansa hakemistonimen ja sen jälkeen poistaa sen:

```
ECHO OFF
IF %1x==x GOTO VIRHE
ECHO Poistan hakemiston %1
ECHO Y | DEL %1\*.*
RD %1
GOTO LOPPU
:VIRHE
ECHO Hakemiston nimi puuttuu!
:LOPPU
```

Komento

```
POISTA ALIHAK
```

antaa tyhjennyskomennon DEL ALIHAK*.* ja välittää sille myönteisen Y-vastauksen. Ilman tätä peräkkäistehtävää poistokäsky pysähtyisi kysymään käyttäjältä kuittausta Are you sure? -kysymykseen. Kun tiedostot on poistettu, komentojono hoitaa vielä hakemistonkin poistamisen RD-komennolla.

POISTA.BATin alkuun on lisätty tarkistus, joka estää komennon ajamisen mikäli ensimmäinen komentoriviparametri puuttuu. Ellei

tarkistusta olisi, poistokomento muuttuisi muotoon DEL *.* ja tyhjentäisi päähakemiston! (Itselleni kävi juuri näin kirjaa tehdessäni). Erityinen huolellisuus ja varovaisuus on paikallaan aina silloin, kun tiedostojen poistokomennon kuittaus kirjoitetaan etukäteen valmiiksi.

Komentojonoa voi vielä tehostaa niin, että ennen poistokäskyä lisätään siirtyminen hakemistoon (CD %1), nollataan kaikkien tiedostojen määreet (ATTRIB -R -H -S *.*) sekä palataan takaisin (CD ..) ennen poistokomennon antamista. Näin tehostettuna komentojono pystyy tyhjentämään myös sellaiset hakemistot, joissa on käytetty suojattuja tiedostoja.

Tämäkään POISTA.BAT ei pysty poistamaan hakemistoa, mikäli sillä on omia alihakemistoja. Työ vaatisi rekursiivista hakemistojen läpikäyntiä ja sitä ei pysty mitenkään helposti ohjelmoimaan komentojonokielellä.

127

Levyn vapaan tilan selvittäminen

"Vapaana" on pieni apuohjelma, joka selvittää paljonko annetulla levyllä on vapaata tilaa ja palauttaa arvon ERRORLEVELin kautta sitä kutsuvalle komentojonolle. Vapaan tilan tietäminen on erityisen hyödyllistä komentojonoissa, joita käytetään ohjelmien asentamiseen tai tiedostojen varmuuskopiointiin. Ohjelma luodaan seuraavasti:

```
N VAPAANA.COM
E 0100 FC BE 81 00 AC 3C 20 74
E 0108 FB 30 D2 3C 0D 74 06 24
E 0110 DF 2C 40 88 C2 B4 36 CD
E 0118 21 F7 E1 F7 E3 B9 14 00
E 0120 D1 EA D1 D8 E2 FA 3D FF
E 0128 00 76 03 B8 FF 00 B4 4C
E 0130 CD 21

RCX
0032
W
Q
```

Sitä käytetään joko kirjoittamalla pelkästään VAPAANA, jolloin ohjelma palauttaa vapaan levytilan määrän oletuslevyllä tai lisäämällä nimen perään halutun levyaseman tunnus, jolloin tulos saadaan miltä levyltä tahansa.

Komentojono, joka varmistaa joukon tiedostoja toiselle levylle (tai lähiverkon läpi serverille) näyttää seuraavalta:

```
ECHO OFF
VAPAANA E:
IF ERRORLEVEL 3 GOTO OK
ECHO Levy E: on lähes täynnä eikä sitä voi
ECHO käyttää varmistukseen!
GOTO LOPPU
:OK
XCOPY *.* E:\VARA /M
:LOPPU
```

Asennusohjelmassa VAPAANA pystyy kertomaan, onko levyllä tarpeeksi vapaata tilaa ohjelmaa varten. Sitä voi käyttää myös kopioinnin jälkeen tutkimaan, jäikö levylle vapaata tilaa. Jos vapaan tilan määrä on alle yhden megan on todennäköistä, että kopiointi on katkennut levytilan täyttymiseen. COPY-käsky ei käytä ERRORLEVELiä, joten siitä ei ole apua kopioinnin tulosta selvitettäessä.

Koska ERRORLEVELin suurin mahdollinen arvo on 255, VAPAANA ei pysty ilmoittamaan tämän suurempia arvoja. Rajoituksella ei kuitenkaan ole käytännön merkitystä.

Merkitystä voi kuitenkin olla sillä, ettei VAPAANA pysty kertomaan mahdollisesta virheestä. Jos sille esimerkiksi annetaan olematon asematunnus se palauttaa arvon nolla, joka voi tarkoittaa että levy on täynnä tai että asematunnus on virheellinen.

128

Tekstitiedoston rivien numerointi

Tekstitiedostojen rivien numerointi tapahtuu näppärästi FINDin avulla. Niksi on siinä, että FINDin käsketään numeroida sellaiset rivit, joilta ei

löydy etsittävää merkkijonoa. Jonoksi valitaan sellainen erikoismerkkien yhdistelmä, jota tiedostosta ei varmasti löydy. Esimerkissä on käytetty jonoa !#$$$&/(, mutta mikä tahansa riittävän mahdoton yhdistelmä kelpaa. Merkkijonossa ei saa kuitenkaan olla varattuja merkkejä kuten %, jota komentotulkki luulee komentojonon sisäiseksi parametrimerkiksi.

Ongelmaksi jää vielä päästä eroon FINDin jokaisen etsinnän alkuun tulostamasta viivasta ja tiedostonimestä, esimerkiksi:

```
---------- ALCHEMY.DOC
```

Tämä tehdään ohjaamalla ensimmäisen FINDin numeroimat rivit toiselle FINDille, joka suodattaa väliviivat pois. Samalla suodattuvat kuitenkin myös mahdolliset tiedostossa olevat väliviivarivit.

Tyhjästä rivistä, jonka FIND tulostaa ennen viivaa ja tiedostonimeä, on mahdotonta päästä eroon. Siksi lopullisessa numeroidussa tiedostossa onkin alussa yksi ylimääräinen tyhjä rivi. Se ei kuitenkaan yleensä tuota ongelmia.

Kerran tehtyä rivien numerointia on erittäin hankala kumota. Ainakaan se ei onnistu pelkällä komentojonolla! Tästä syystä varmistetaan alkuperäinen tiedosto nimelle VARATIED.DDD ennen kuin numerointi tehdään. Lopullinen komentojono näyttää seuraavalta:

```
ECHO OFF
COPY %1 VARATIED.DDD >NUL
COPY %1 TEMP.TMP >NUL
FIND /N/V "!#$$$&/(" TEMP.TMP|FIND /V "-------" >%1
DEL TEMP.TMP
ECHO Tiedosto %1 numeroitu,
ECHO alkuperäinen nimellä VARATIED.DDD
```

129

Rivien poistaminen tiedostosta

FINDin avulla voi myös poistaa haluttuja rivejä tiedostosta, kunhan tiedosto vain on puhdasta ASCIIta. Esimerkiksi Telix-tietoliikenneohjelma kirjoittaa levylle käytön aikana lokitiedostoa TELIX.USE, johon kirjaantuvat kaikki otetut yhteydet, niiden kestot, käytetyt asetukset sekä siirrettyjen tiedostojen nimet. Ahkerassa käytössä tiedosto paisuu nopeasti satoihin kilotavuihin, jolloin turhat rivit kannattaa poistaa tilan säästämiseksi.

Eräs turha rivi on lokitiedoston avauksesta kertova merkintä

```
93-01-18  09:32:46  Telix Usage Log Opened.
```

joka syntyy aina kun ohjelma käynnistetään. Rivi poistetaan komentojonolla seuraavasti:

```
ECHO OFF
TYPE TELIX.USE|FIND /V "Usage Log Opened" >APU.$$$
IF NOT EXIST APU.$$$ GOTO LEVYTÄYNNÄ
DEL TELIX.USE
REN APU.$$$ TELIX.USE
GOTO LOPPU
:LEVYTÄYNNÄ
ECHO Rivien poistoa ei voitu tehdä
ECHO koska levy täyttyi
:LOPPU
```

Jos poistettavia tekstejä on useita, ne voidaan kirjoittaa joko yhdeksi pitkäksi putkeksi (FIND /V "Usage Log Opened" | FIND /V "Usage Log Closed" | FIND /V "tähän kolmas ehto jne." >APU.$$$) tai sitten ajaa useammassa vaiheessa, jolloin jokaisella kierroksella poistetaan yhden ehdon täyttävä rivi.

130

Viikonpäivän selvittäminen komentojonossa

Komentojonolle, joka osaisi itse tutkia mikä viikonpäivä on kyseessä, olisi helppo löytää monia sovelluskohteita. Se voisi esimerkiksi tervehtiä käyttäjää maanantaiaamuna ja muistuttaa perjantaina siitä, että tänään olisi syytä varmistaa viikon aikana kertyneet tiedostot.

Vahinko vain, ettei komentojonokielessä ole omaa komentoa päivän selvittämiseksi. Sellainen on kuitenkin mahdollista toteuttaa kikkailemalla.

Ensiksi kysytään DATE-käskyltä, mikä päiväys tänään on. DATE esittää päiväyksen muodossa

```
Current date is Sun 26.01.1992
Enter new date:
```

ja jää odottamaan uutta päiväystä. Kuittaus luetaan tiedostosta ja päiväyksestä päästetään läpi vain Currentista kertova rivi. Se ohjataan aputiedostoon TEMP.BAT:

```
DATE <CR.CR | FIND "Current" >TEMP.BAT
```

Nyt TEMP.BATissa lukee DATE-käskyn antama rivi. Kun se suoritetaan, komentotulkki luulee Current-sanaa nimeksi ja loppuosaa rivistä sen parametriksi. Niinpä luodaan levylle yksinkertainen komentojono CURRENT.BAT, joka ottaa kolmannen parametrin talteen, sijoittaa sen ympäristömuuttujaan PAIVA ja poistaa lopuksi TEMP.BAT -aputiedoston:

```
SET PAIVA=%3
DEL TEMP.BAT
```

Tämän jälkeen päivän englanninkielinen nimi löytyy ympäristömuuttujasta PAIVA ja sitä voidaan tutkia muissa komentojonoissa käyttämällä merkintää %PAIVA%.

Kokonaisuudessaan tarvitaan siis kaksi komentojonoa:

```
HAEPAIVA.BAT:
DATE <CR.CR | FIND "Current" >TEMP.BAT
TEMP
```

sekä

```
CURRENT.BAT:
SET PAIVA=%3
DEL TEMP.BAT
```

Komentojonon käyttö on kuitenkin hieman hankalaa, koska sitä ei voida kutsua aliohjelmana CALL HAEPAIVA. Päivän haku kannattaakin laittaa AUTOEXEC.BATin loppuun tai sitten saatua päiväarvoa tutkiva komentojono käynnistetään CURRENT.BATin viimeisellä rivillä. Se voi näyttää vaikkapa seuraavalta:

```
ECHO OFF
ECHO Tänään on...
IF %PAIVA%==Mon ECHO Maanantai
IF %PAIVA%==Tue ECHO Tiistai
IF %PAIVA%==Wed ECHO Keskiviikko
IF %PAIVA%==Thu ECHO Torstai
IF %PAIVA%==Fri ECHO Perjantai
IF %PAIVA%==Sat ECHO Lauantai
IF %PAIVA%==Sun ECHO Sunnuntai
```

Jos päiväystietoa tarvitaan, HAEPAIVA.BATin kutsu voidaan lisätä jo AUTOEXEC.BATiin jolloin päivän hakemista ei tarvitse tehdä joka kerta kun sitä halutaan käyttää.

Jos toimintovaihtoehtoja on paljon, komentojonoa voidaan lyhentää kirjoittamalla

```
GOTO %PAIVA%
:MON
REM tähän maanantain komennot
....
GOTO LOPPU
:TUE
REM tähän tiistain komennot
...
```

```
GOTO LOPPU
:WED
REM tähän keskiviikon komennot
....
jne
:LOPPU
```

Mikäli käytetään suomennettua DOSia, CURRENT.BAT ja viikonpäivien nimet pitää vaihtaa käännöksen mukaisesti.

Kellonajan selvittäminen onnistuu samalla periaatteella mutta jos tarvitaan sekä päivä- että kellonaikatietoa, voidaan molemmat poimia DIR-listauksesta seuraavan NYT.BAT-komentojonon avulla:

```
ECHO OFF
IF NOT %4X==X GOTO OK
COPY /B NYT.BAT+,,
DIR NYT.BAT | FIND "NYT" >TEMP.BAT
TEMP
:OK
SET PVM=%3
SET AIKA=%4
DEL TEMP.BAT
ECHO AIKA = %AIKA%
ECHO PÄIVÄ= %PVM%
```

Komentojono tallentaa päiväyksen ja kellonajan vastaaviin ympäristömuuttujiin DOSin esittämässä muodossa.

Voisiko viikonpäivän selvittää helpomminkin? Toki. Seuraava SRC-listaus luo levylle tiedoston HAE-PV.COM, joka palauttaa viikonpäivän numeron ERRORLEVEL-paluukoodina.

```
N HAE-PV.COM
A 0100
MOV  AH,2A
INT  21
MOV AH,4C
INT 21

RCX
0008
W
Q
```

Päivän selvittäminen tapahtuu nyt seuraavasti:

```
HAE-PV
IF ERRORLEVEL 6 GOTO LAUANTAI
IF ERRORLEVEL 5 GOTO PERJANTAI
jne.
```

131

Nollatiedoston luominen

Ympäristömuuttujien lisäksi myös tiedostoja voidaan käyttää yksinkertaisina muisteina. Ne kertovat, onko jokin asia tapahtunut vai ei. Ympäristömuuttujiin verrattuna tällaisilla muistitiedostoilla on se etu, että ne toimivat myös ajettaessa DOS-ohjelmia moniajossa Windowsin tai OS/2:n alla sekä lähiverkossa, kun samalla hakemistolla voi olla useita yhtäaikaisia käyttäjiä.

Jos esimerkiksi halutaan panna muistiin, että jokin asia on jo tapahtunut, levylle luodaan siitä muistuttava tiedosto (esim. ONJO.TMP). Kun asiaa halutaan tutkia, käytetään IF EXIST komentoa. Muistutus poistetaan tavallisella DEL-käskyllä.

Mutta miten tehdä mahdollisimman pieni muistilapputiedosto? Kuten niksissä 85 kävi ilmi, jokainen tiedosto vie levyltä vähintään yhden varausyksikön. Vaikka tiedoston pituus olisi vain yksi tavu, se kuluttaa kiintolevyllä vähintään kaksi kilotavua. Vasta kun tiedoston pituudeksi saadaan nolla, käyttöjärjestelmä ei varaa sille tilaa lainkaan.

Nollatiedoston tekeminen onnistuu REM-lauseen ja ohjausmerkin avulla. Komento

```
REM >ONJO.TMP
```

luo nolla tavua pitkän tiedoston nimelle ONJO.TMP. Muisteina toimivat tiedostot kannattaa nimetä TMP:ksi, jotta ne olisi helppo löytää ja poistaa mikäli komentojono ei jostain syystä itse tee sitä.

Kun tiedosto on luotu, sitä voidaan tutkia IF EXIST-lauseella seuraavasti:

```
IF EXIST ONJO.TMP ECHO On tehty jo!
```

131 B

Nollatiedostoa ei voida kopioida COPY-käskyllä, sillä COPY ilmoittaa "0 file(s) copied". Tätä ominaisuutta voidaan käyttää hyväksi, kuten niksissä 152 ilmenee. Jos nollatiedosto kuitenkin halutaan kopioida sellaisenaan, pitää käyttää XCOPYä.

Merkkitiedoston avulla saadaan välitettyä tietoa jopa koneelta toiselle. Tästä voi olla suurta hyötyä varsinkin lähiverkoissa, kuten seuraava niksi osoittaa.

132

Ohjelman varaus verkossa

Jos lähiverkkoon on asennettu sovellus, joka ei ole varsinainen verkkoversio ja johon ei ole ostettu kuin yksi käyttöoikeus, voidaan edellisen niksin avulla luoda komentojono, joka varmistaa että sovellusta voi käyttää vain yksi työasema kerrallaan.

Ohjelma otetaan käyttöön komentojonolla, joka ensin tutkii, onko jokin toinen työasema ehtinyt jo varata ohjelman itselleen:

```
IF EXIST X:KAYTOSSA.TMP GOTO EIKAY
```

X: tarkoittaa asematunnusta, johon kaikilla verkon käyttäjillä on luku- ja kirjoitusoikeus. Koska sovellusten hakemistot pitäisi jakaa pelkillä kirjoitusoikeuksilla, tämän X: aseman pitäisi olla eri kuin sovelluksen hakemisto.

Ellei kukaan ole varannut ohjelmaa, tehdään se itse:

```
REM >X:KAYTOSSA.TMP
```

ja annetaan tämän jälkeen käynnistämiseen tarvittavat komennot. Kun ohjelma päättyy, poistetaan KAYTOSSA.TMP merkiksi muille, että ohjelma on jälleen vapaa. Mikäli jaettavia ohjelmia on useita, niiden käyttämät merkkitiedostot kannattaa nimetä ohjelman nimen mukaisesti päällekkäisyyksien välttämiseksi.

Jos ohjelma oli käytössä, hypätään kohtaan EIKAY, missä neuvotaan käyttäjää yrittämään myöhemmin uudelleen:

```
:EIKAY
ECHO Ohjelmaa käytetään parhaillaan. Yritä
ECHO myöhemmin uudelleen!
:LOPPU
```

Kun tähän vielä lisätään verkkoyhteyden avaamiseen ja sulkemiseen tarvittavat komennot, saadaan lopulliseksi komentojonoksi:

```
ECHO OFF
REM testataan onko ohjelma käytössä
IF EXIST X:KAYTOSSA.TMP GOTO EIKAY
REM Ei ole, varataan se itselle
REM >X:KAYTOSSA.TMP
REM Sitten avataan resurssi
NET USE Y:  \\SERVERI\SOFTA
REM ja käynnistetään sovellus
Y:SOFTA
REM kun se loppuu, poistetaan merkki
DEL X:KAYTOSSA.TMP
REM ja suljetaan resurssi
NET USE Y: /D
ECHO Ohjelma on nyt muiden käytettävissä
GOTO loppu
:EIKAY
ECHO Ohjelmaa käytetään parhaillaan. Yritä
ECHO hetken päästä uudelleen!
:LOPPU
```

Samalla periaatteella on mahdollista sallia ohjelman käyttö esimerkiksi kahdelle tai enintään kolmelle yhtä aikaiselle käyttäjälle.

133

Ohjelman ajo kerran päivässä

Edellisiä niksejä yhdistelemällä saadaan aikaan komentojono, joka ajaa ohjelman kertaalleen annettuna viikonpäivänä. Esimerkiksi virusten etsintä tai TEMP-hakemiston tyhjennys saadaan tapahtumaan vain maanantain ensimmäisellä käynnistyskerralla.

Oletetaan, että päivän numero on haettu tavalla tai toisella ympäristömuuttujaan PV ja eri päiviä kuvataan kokonaisluvuilla niin, että maanantaita vastaa arvo 1. Ensin testataan, onko maanantai. Jos on, tutkitaan onko ajamisesta kertova merkkitiedosto jo olemassa. Ellei ole, tiedosto luodaan ja haluttu ajo tehdään. Mikäli komentojono ajetaan samana päivänä uudelleen, merkkitiedosto kertoo ettei ajoa tarvita.

Mikäli päivä poikkeaa halutusta, mahdollinen merkkitiedosto poistetaan. IF-lause estää File not found-virheilmoituksen tulostumisen ja komentojono kokonaisuudessaan näyttää seuraavalta:

```
IF NOT %PV%==1 GOTO MUUPAIVA
REM on maanantai, tarkistetaan onko jo ajettu
IF EXIST \TEMP\MAANANT.MRK GOTO LOPPU
REM ei ole, joten luodaan merkkitiedosto...
REM >\TEMP\MAANANT.MRK
REM ja tehdään se, mikä mikron täytyy
ECHO On maanantain ensimmäinen käynnistyskerta
ECHO joten ajan nyt virustarkistuksen, odota...
SCAN C:
IF ERRORLEVEL 1 CALL HALYTYS.BAT
GOTO LOPPU
:MUUPAIVA
IF EXIST \TEMP\MAANANT.MRK DEL \TEMP\MAANANT.MRK
:LOPPU
```

134

Ajo joka n:s kerta

Edellistä niksiä soveltamalla saadaan tehtyä komentojono, joka käynnistää halutun ohjelman esimerkiksi joka kolmannella tai viidennellä kerralla. Laskurina käytetään tiedostoa KERTA.X, missä X on ajokerran numero. Tiedoston numeroa tutkitaan ja sitä kasvatetaan yhdellä jokaisella käynnistyskerralla. Kun haluttu arvo on saavutettu, tiedosto poistetaan jolloin seuraava ajokerta aloittaa laskemisen alusta. Komentojono on hyvin yksinkertainen, mutta sitä työläämpi kirjoittaa mitä useampia kierroksia halutaan kuluvan ajokertojen välillä.

Seuraava jono TARKISTA.BAT käynnistää SCAN-komentojonon joka viidennellä ajokerralla. Kun AUTOEXEC.BATiin lisätään komento CALL TARKISTA.BAT, tarkistus tulee suoritettua automaattisesti joka viidennellä käynnistyksellä.

```
IF NOT EXIST KERTA.* GOTO NOLLA
IF EXIST KERTA.1 GOTO YKSI
IF EXIST KERTA.2 GOTO KAKSI
IF EXIST KERTA.3 GOTO KOLME
IF EXIST KERTA.4 GOTO NELJA
IF EXIST KERTA.5 GOTO VIISI
:NOLLA
REM >KERTA.1
GOTO LOPPU
:YKSI
REN KERTA.1 KERTA.2
GOTO LOPPU
:KAKSI
REN KERTA.2 KERTA.3
GOTO LOPPU
:KOLME
REN KERTA.3 KERTA.4
GOTO LOPPU
:NELJA
REN KERTA.4 KERTA.5
:VIISI
DEL KERTA.5
SCAN C:
:LOPPU
```

Vastaavalla tavalla voi hoitaa esimerkiksi TEMP-hakemiston tyhjennyksen joka toisella tai kolmannella käynnistyskerralla. Näin käyttäjälle jää mahdollisuus tutkia edelliseltä käyttökerralta jääneitä aputiedostoja siltä varalta, että niistä voitaisiin vielä palauttaa jotakin hyödyllistä.

135

ERRORLEVEL-arvon tutkiminen

DOSissa oleva ERRORLEVEL-paluukoodi on ainoa tapa, millä ohjelmat pystyvät palauttamaan tietoa niitä kutsuneille komentojonoille. Siksi kannattaa oppia käyttämään ERRORLEVELiä mahdollisimman tehokkaasti.

Ohjelman palauttamaa ERRORLEVEL-arvoa ei voida tutkia suoraan IF ERRORLEVEL==x vaan kirjoittamalla

```
IF ERRORLEVEL x
```

joka toteutuu aina, kun paluukoodi on x tai suurempi. Vertailun tekeminen tällä tavoin saattaa näyttää kömpelöltä, mutta se riittää useimpiin tapauksiin sillä yleensä halutaan vain tietää, onko ajo sujunut normaalisti (jolloin ERRORLEVEL palauttaa nollan) vai ei. Rivi

```
IF ERRORLEVEL 1
```

kertoo, että paluukoodi on ollut suurempi kuin nolla ja jotain outoa on sattunut.

Jos paluukoodin arvo halutaan selvittää tarkemmin, on numeroiden vertailu tehtävä isommasta pienempään, esimerkiksi seuraavasti:

```
IF ERRORLEVEL 5 GOTO ERVIISI
IF ERRORLEVEL 4 GOTO ERNELJA
IF ERRORLEVEL 3 GOTO ERKOLME
IF ERRORLEVEL 2 GOTO ERKAKSI
IF ERRORLEVEL 1 GOTO ERYKSI
```

Koska ERRORLEVELin selvittäminen tätä kautta on työlästä, kannattaa luoda aliohjelmana käytettävä komentojono, joka siirtää

ERRORLEVEL-arvon ympäristömuuttujaan EL. Tarvittavien portaiden määrästä riippuen komentojono MUUNNAEL.BAT voisi näyttää esimerkiksi seuraavalta:

```
ECHO OFF
SET EL=0
IF ERRORLEVEL 1 SET EL=1
IF ERRORLEVEL 2 SET EL=2
IF ERRORLEVEL 3 SET EL=3
IF ERRORLEVEL 4 SET EL=4
IF ERRORLEVEL 5 SET EL=5
IF ERRORLEVEL 6 SET EL=6
IF ERRORLEVEL 7 SET EL=7
```

Tässä tapauksessa arvojen vertailu pitää tehdä pienimmästä suurimpaan, jotta viimeinen toteutuva sijoituslause jäisi voimaan.

Komentojonoa voidaan kutsua CALL-lauseella toisessa jonossa annetun ohjelmakäskyn jälkeen seuraavasti:

```
ECHO OFF
XCOPY *.* E:\VARA /M
CALL MUUNNAEL.BAT
IF %EL%==0 ECHO Kopiointi sujui OK.
IF %EL%==1 ECHO Ei mitään kopioitavaa!
IF %EL%==2 ECHO Ohjelma keskeytetty!
IF %EL%==4 ECHO Virhe komennon muodossa
IF %EL%==5 ECHO Levyvirhe!
```

Jos tutkittavia arvoja ja niiden perusteella tehtäviä toimia on paljon, kannattaa käyttää muotoa

```
GOTO %EL%
```

ja käsitellä kukin tapaus erikseen.

Jos taas arvot jakautuvat usealle eri välille eli halutaan testata onko

```
x<= ERRORLEVEL < y
```

voidaan käyttää kahta peräkkäistä IF-lausetta seuraavasti:

```
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 6 GOTO 05
IF ERRORLEVEL 6 IF NOT ERRORLEVEL 12 GOTO 0611
```

```
:05
REM arvo oli nollasta viiteen
GOTO LOPPU
:0611
REM arvo oli kuudesta yhteentoista
:LOPPU
```

Tästä menettelystä on hyötyä seuraavassa niksissä, jossa tutkitaan kellonaikaa.

Saman niksin muunnoksena saadaan vielä tapaus, jossa halutaan testata onko ERRORLEVELillä jokin tietyistä arvoista silloin, kun tutkittava arvoalue ei ole jatkuva. Komennot

```
IF ERRORLEVEL 10 IF NOT ERRORLEVEL 11 ECHO oli 10
IF ERRORLEVEL 20 IF NOT ERRORLEVEL 21 ECHO oli 20
IF ERRORLEVEL 30 IF NOT ERRORLEVEL 31 ECHO oli 30
```

tutkivat, onko ERRORLEVEL-arvo joko 10, 20 tai 30.

136

Kellonajan lukeminen

Seuraava pieni ohjelma palauttaa koneen kellossa olevan tuntilukeman ERRORLEVELinä takaisin kutsuvalle komentojonolle:

```
N HAETUNTI.COM
E 0100 B4 2C CD 21 88 E8 B4 4C
E 0108 CD 21
RCX
000A
W
Q
```

Esimerkiksi peliohjelman käynnistäminen verkossa voidaan rajoittaa tapahtuvaksi vain työajan jälkeen seuraavalla PELI.BAT-komentojonolla:

```
ECHO OFF
```

```
HAETUNTI
IF ERRORLEVEL 16 GOTO OK
ECHO Peliä voi pelata ainoastaan klo 16 jälkeen!
GOTO LOPPU
:OK
ECHO Peli käynnistyy, odota hetki...
NET USE P: \\SEVERI\PELIT
P:
PELI
NET USE P: /D
:LOPPU
```

Tällainen suojaus ei tietenkään ole kovin tehokas, sillä vähänkin DOSia tunteva käyttäjä voi muuttaa kellonaikaa TIME-komennolla ennen PELI.BATin käynnistymistä.

Jos halutaan tutkia, onko kellonaika annetulla välillä, pitää käyttää kohdassa 135 esitettyä tapaa ERRORLEVEL-arvon vertailuun. Seuraava komentojono esittää vaihtelevan tervehdyksen ajoajasta riippuen:

```
ECHO OFF
HAETUNTI
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 7 GOTO YO
IF ERRORLEVEL 7 IF NOT ERRORLEVEL 11 GOTO AAMU
IF ERRORLEVEL 11 IF NOT ERRORLEVEL 18 GOTO PAIVA
IF ERRORLEVEL 18 GOTO ILTA
:YO
ECHO Hyvää yötä!
GOTO LOPPU
:AAMU
ECHO Hyvää huomenta!
GOTO LOPPU
:PAIVA
ECHO Hyvää päivää!
GOTO LOPPU
:ILTA
ECHO Hyvää iltaa!
:LOPPU
```

137

Onko alihakemisto tai levyasema olemassa?

Tiedoston olemassaolo voidaan selvittää IF EXIST-lauseella seuraavasti:

```
IF EXIST JUTTU.TXT ECHO Tiedosto on olemassa
```

Samaa vertailukäskyä ei kuitenkaan voida käyttää, mikäli testin kohteena on alihakemistonimi, vaan testi pitää kirjoittaa muodossa

```
IF EXIST \ALIHAK\NUL
```

Kysymys palauttaa arvon tosi, jos hakemisto on olemassa riippumatta siitä, onko siinä tiedostoja. Jos halutaan vielä selvittää tiedostojenkin olemassaolo, kirjoitetaan

```
IF EXIST \ALIHAK\*.*
```

137 B

Levyaseman tarkistus tapahtuu samalla periaatteella. Seuraava jono luettelee käytettävissä olevat asematunnukset väliltä D-G:

```
IF EXIST D:\NUL ECHO Asema D: on
IF EXIST E:\NUL ECHO Asema E: on
IF EXIST F:\NUL ECHO Asema F: on
IF EXIST G:\NUL ECHO Asema G: on
```

Tästä tarkistuksesta on hyötyä varsinkin silloin, kun komentojonoa käytetään ohjelman asentamiseen kiintolevylle. Asennus aloitetaan vasta kun on selvitetty, että levyllä todella on käyttäjän ilmoittama looginen asematunnus johon tiedostot sitten kopioidaan.

138

ECHO ja tyhjä rivi

Tekstiä kuvaruudulle tulostava ECHO vaatii pienen niksin ennen kuin se suostuu tulostamaan tyhjän rivin. Pelkkä komento ECHO tulostaa joko Echo off tai Echo on, riippuen siitä onko kaiutus päällä tai ei, joten siitä ei ole hyötyä.

DOS 3.0:ssa tyhjä rivi saadaan kirjoittamalla ECHO ja kaksi peräkkäistä välilyöntiä. Uudemmissa versioissa on kuitenkin käytettävä muita keinoja. Eräs klassinen tapa on käyttää hyödyllistä tyhjämerkkiä ja kirjoittaa ECHO <255>, missä <255> on yhdessä Alt-näppäimen kanssa kirjoitettava numerokoodi.

Toinen, hieman elegantimpi tapa on kirjoittaa

```
ECHO.
```

joka toimii DOS-versiosta 3.1 alkaen. Pisteen paikalla voi myös olla kenoviiva, kauttaviiva, plussa, lainausmerkki tai kaksoispiste. Vakiintunut tapa on kuitenkin käyttää juuri pistettä. DR-DOSeissa 5.0 ja 6.0 riittää ECHO ja vähintään kaksi välilyöntiä sen perään.

139

Ilmoitusten piilottaminen

Komentojonon alussa oleva ECHO OFF estää komentojen nimien tulostumisen ("kaiuttamisen") kuvaruudulle jonon ajonaikana. Se ei kuitenkaan estä komentojen tuottamia ilmoituksia näkymästä. Esimerkiksi tiedostojen kopioinnin aikana ruudulle tulostuu

```
1 file(s) copied
```

Näistä ilmoituksista on helppo päästä eroon ohjaamalla ilmoitukset mustaan aukkoon eli NULliin. Kun COPY-käsky kirjoitetaan muodossa

```
COPY JUTTU.TXT D:\VARA >NUL
```

"files copied" -tekstiä ei näy.

Valitettavasti ei ole mitään tapaa jolla voisi estää virheilmoitusten näkymisen. Komento

```
DEL TURHA.APU
```

tulostaa aina ruudulle varoituksen File not found. Ainoa tapa päästä eroon virheilmoituksista on estää niiden syntyminen. DEL-käskyn yhteydessä se on helppoa esimerkiksi komennolla

```
IF EXIST TURHA.APU DEL TURHA.APU
```

mutta mitään yleispätevää, kaikille komennoille sopivaa menettelyä ei ole.

140

Komentojen toistaminen

FOR-lauseen avulla on helppo toistaa komentoa useita kertoja peräkkäin. Niksi perustuu siihen, että FOR-lauseen indeksillä ei tarvitse olla mitään käyttöä toistettavassa komennossa. Se voi toimia pelkkänä laskurina.

Esimerkiksi viisi tyhjää riviä tulostuu komentojonossa helpoimmin komennolla

```
FOR %%i IN (1 2 3 4 5) DO ECHO.
```

Vastaavasti ruudun tyhjennys ja DIR-listaus tapahtuu komentoriviltä annettuna seuraavasti (valitettavasti valitsimia tms. ei voi käyttää):

```
C:\>FOR %i IN (CLS DIR) DO %i
```

141

Paitsi.bat

DOSista puuttuu toiminto, joka rajaisi osan tiedostoista seuraavan komennon ulkopuolelle. DOS 5.0:sta alkaen tällainen toiminto voidaan kuitenkin luoda komentojonon avulla. Komentojonon toimintaa kuvaa hyvin sen nimi PAITSI.BAT.

Tiedostojen jättäminen komennon ulkopuolelle perustuu niiden piilottamiseen ATTRIB +H komennolla. Kun tiedostot on piilotettu, loppuosa komentorivistä välitetään komentotulkille normaaliin tapaan.

Komentojono näyttää kaikessa yksinkertaisuudessaan seuraavalta:

```
ECHO OFF
ATTRIB +H %1
%2 %3 %4 %5 %6 %7 %8
ATTRIB -H %1
```

Lyhyydestään huolimatta jonolla on monta käyttöä. Esimerkiksi komento

```
PAITSI *.DOC DEL *.*
```

poistaa oletushakemistosta kaikki muut paitsi DOC-tiedostot. Vastaavasti komento

```
PAITSI *.BAK COPY *.* A:
```

kopioi oletushakemiston tiedostot BAKeja lukuunottamatta A: levykkeelle.

Koska komentojono toimii vain DOS 5.0:sta alkaen sen käyttämän ATTRIB -H komennon vuoksi, jonon alkuun kannattaisi lisätä niksissä 143 esitetty DOS-version tarkistus. Muuten käy niin, että vanhemmalla DOS-versiolla piilotusta ei tapahdu ja DEL *.* poistaa kaikki hakemiston tiedostot.

142

Merkin lukeminen näppäimistöltä

Komentojonokielessä on sen yksinkertaisuudesta huolimatta kaikki oikean ohjelmointikielen tunnusmerkit — vain mahdollisuus lukea näppäimistöä puuttuu. Tämä puute saatiin korjattua vasta 6.0-versiossa.

Vanhemmissa DOS-versioissa yksinkertainen INPUT-lause on helppo ohjelmoida omin päin seuraavasti:

```
N KEY.COM
E 0100 B4 00 CD 16 3C 7A 77 06
E 0108 3C 60 76 02 24 DF 3C 00
E 0110 75 02 88 E0 B4 4C CD 21
RCX
0018
W
Q
```

Kun KEYtä kutsutaan komentojonossa se jää odottamaan näppäimen painallusta ja palauttaa painetun näppäimen ASCII-arvon ERRORLEVELinä. Jos halutaan testata tiettyä näppäintä pitää muistaa, että isoilla ja pienillä kirjaimilla on eri ASCII-koodi ja että ERRORLEVELin luonteen vuoksi testaus pitää aloittaa isoimmasta arvosta alkaen.

Nämä seikat huomioiden saadaan kirjoitettua asennusohjelma, joka kysyy käyttäjältä haluaako hän jatkaa. Ison K:n ASCII-koodi on 75 ja pienen 107. E-kirjaimella vastaavat koodit ovat 69 ja 101. Tämä osa asennusjonosta näyttää seuraavalta:

```
:EIKELPAA
ECHO Haluatko jatkaa asennusta? (K=kyllä, E=ei)
KEY
IF ERRORLEVEL 107 IF NOT ERRORLEVEL 108 GOTO OK
IF ERRORLEVEL 75 IF NOT ERRORLEVEL 76 GOTO OK
IF ERRORLEVEL 101 IF NOT ERRORLEVEL 102 GOTO EI
IF ERRORLEVEL 69 IF NOT ERRORLEVEL 70 GOTO EI
REM jos ei mikään näistä niin uusi kysymys
GOTO EIKELPAA
:OK
REM haluttiin jatkaa, komennot jatkuvat tästä
```

```
:EI
REM ei haluttu jatkaa, siis lopetus tähän
```

KEY-ohjelma toimii myös yksinkertaisena käynnistysvalikkona. Ruudulle tulostetaan ensiksi valikko

```
CLS
ECHO  Valitse ajettavan ohjelman numero:
ECHO.
ECHO 1 - Tekstinkäsittely (WP)
ECHO 2 - Tietoliikenne-yhteys pankkiin (Telix)
ECHO 3 - Taulukkolaskenta (Lotus)
```

ja sen jälkeen tutkitaan, minkä vaihtoehdon käyttäjä haluaa:

```
:EIKAY
KEY
REM tutkitaan ASCII-merkki "3" (koodi 51)
IF ERRORLEVEL 51 GOTO AJALOTUS
REM tutkitaan "2" (koodi 50)
IF ERRORLEVEL 50 GOTO AJATELIX
REM tutkitaan "1" (koodi 49)
IF ERRORLEVEL 49 GOTO AJAWP
GOTO EIKAY
```

143

DOS-version selvittäminen

Komentojonokielen yksityiskohdat vaihtelevat hieman riippuen siitä, mikä DOS-versio on käytössä. Joskus voi olla tärkeätä tietää myös se, toimitaanko "aidossa" DOSissa vai onko kyseessä OS/2:n DOS-tila, jolloin moniajossa voi olla muitakin sovelluksia eikä esimerkiksi koneen ohjelmallista uudelleenkäynnistystä (buuttausta) pidä mennä tekemään.

Käytössä olevan DOSin versionumero saadaan selville seuraavalla pikku ohjelmalla:

```
N DOSVERS.COM
E 0100 B4 30 CD 21 B4 4C CD 21
```

```
RCX
8
W
Q
```

Ohjelma palauttaa ERRORLEVEL-arvona DOSin version kokonaisosan. Tätä voidaan käyttää esimerkiksi seuraavasti:

```
ECHO OFF
DOSVERS
IF ERRORLEVEL 4 GOTO NELJÄ
IF ERRORLEVEL 3 GOTO KOLME
:NELJÄ
ECHO versio on 4.x tai uudempi
GOTO LOPPU
:KOLME
ECHO versio on 3.x
:LOPPU
```

Versionumerot voivat myös tuottaa yllätyksiä. Esimerkiksi DR-DOSin versiot 5.0 ja 6.0 palauttavat kutsujalle arvon 3.3. Tällä DR-DOSin tekijät ovat halunneet välttää tilannetta, joka aluksi vaivasi DOS 4.0:aa, kun eräät sovellukset kieltäytyivät toimimasta nähdessään tavallista isomman DOS-version.

OS/2:n DOS-tila on helppo tunnistaa siitä, että versionumero on yli 10. Esimerkiksi OS/2:n ensimmäiset 1.x-versiot antavat numeron 10 tai 11 ja 2.0-versio numeron 20. Seuraava komentojono kertoo, ollaanko DOSissa vai OS/2:ssa:

```
ECHO OFF
DOSVERS
IF ERRORLEVEL 10 GOTO OS2
ECHO Tämä on perus-DOS
GOTO LOPPU
:OS2
ECHO Tämä on OS/2
:LOPPU
```

144

PATHin käsittely komentojonossa

PATHissa olevat hakemistonimet on helppo poimia erilleen lähempää tutkimista varten, sillä DOS kohtelee hakemistonimiä erottavaa puolipistettä välilyönnin tapaan. Komentojono

```
ECHO OFF
FOR %%i IN (%PATH%) DO ECHO %%i
```

tulostaa PATHissa olevat hakemistot yksi kerrallaan allekkain ruudulle. Ongelma on vain siinä, että jos PATH on pitkä, se komentojonoon sijoitettuna tekee rivistä liian pitkän eikä tulostus enää PATHin viimeisille hakemistoille.

Usein syntyy tarve testata, löytyykö annetun niminen tiedosto jostakin polun varrella olevasta hakemistosta. Ensimmäinen yritys kirjoittaa komentojono ONKOPATH.BAT näyttää seuraavalta:

```
FOR %%i IN (%PATH%) DO IF EXIST %%i\%1 ECHO Löytyi
```

Kun kirjoitetaan

```
ONKOPATH WP.EXE
```

se tulostaa Löytyi, jos WP.EXE-tiedosto löytyy jostakin PATHin osoittamasta hakemistosta. Sitä voi käyttää myös korvausmerkeillä, kuten

```
ONKOPATH *.BAK
```

joka antaa myönteisen vastauksen mikäli jostakin PATHin hakemistosta löytyy BAK-tiedostoja.

Komentojonossa on kuitenkin edelleen rivin pituudesta aiheutuva ongelma. Mikäli PATH on liian pitkä, FOR-rivin loppu katkeaa eikä etsintä enää toimi luotettavasti. Tätä ongelmaa ei saada poistettua kokonaan, mutta melko hyvään tulokseen päästään jakamalla komentojono kahteen osaan ja välittämällä etsittävän tiedoston nimi ympäristömuuttujassa komentorivin sijaan.

Etsinnän käynnistävä ONKOPATH.BAT-jono näyttää nyt seuraavalta:

```
ECHO OFF
SET loytyi=0
SET etsittava=%1
```

Muuttuja loytyi asetetaan nollaksi ja etsittävän tiedoston nimi sijoitetaan etsittava-nimiseen muuttujaan. Sen jälkeen käynnistetään etsintäjono O.BAT, jolle välitetään PATH ikään kuin se olisi komentoriviparametri:

```
CALL O %path%
```

Etsintäjono O.BAT ottaa saamansa polun ja purkaa sen tekijöihinsä. Jokaisen hakemistonimen kohdalla etsitään ympäristömuuttujassa annettua tiedostoa ja jos se löytyy, ympäristömuuttuja loytyi saa arvon yksi:

```
ECHO OFF
:ALKU
IF %1x==x GOTO loppu
REM tutkitaan hakemistoa %1
IF EXIST %1\%etsittava% set loytyi=1
REM ...tai ECHO tiedosto %etsittava% on
REM hakemistossa %1
SHIFT
GOTO ALKU
:LOPPU
```

Kun PATHin osat on tutkittu, palataan takaisin kutsuvaan jonoon, joka kokonaisuudessaan näyttää seuraavalta:

```
ECHO OFF
SET loytyi=0
SET etsittava=%1
CALL O %path%
IF %loytyi%==1 ECHO Tiedosto löytyi PATHin varrelta
IF %loytyi%==0 ECHO Tiedostoa ei löytynyt PATHista
REM vapautetaan muuttujat muistin säästämiseksi
SET loytyi=
SET etsittava=
```

Etsintä käynnistyy komentojonolla jota kutsutaan komennolla CALL O. Tämä komento on vain hieman pidempi kuin hakupolun määrittelevä PATH=, joten polku saadaan käsiteltyä oikein vaikka se olisi pitkäkin. Jos polku on määritelty enimmäispituuteensa, komentojono jättää siitä viimeiset kolme merkkiä pois. Tällainen tilanne on kuitenkin niin harvinainen, ettei se käytännössä rajoita komentojonon käytettävyyttä.

145

Sovellusten käynnistäminen komentojonoilla

Sen sijaan, että sovellusten käynnistystä helpotettaisiin sijoittamalla kaikki sovellushakemistot PATHiin, kannattaa sovellukset mieluummin käynnistää komentojonojen avulla. Jokaista tärkeätä sovellusta varten kirjoitetaan oma käynnistysjononsa, joka sijoitetaan vaikkapa C:\BATS -hakemistoon. Sen jälkeen tämä hakemisto lisätään PATHiin.

Esimerkiksi WordPerfectin käynnistysjono WP.BAT voisi näyttää seuraavalta:

```
ECHO OFF
ECHO Käynnistän WP:n, odota hetki...
D:
CD \WP\TYOT
SET VANHA=%PATH%
PATH=D:\WP;C:\DOS;C:\UTIL;C:\BATS
WP
SET PATH=%VANHA%
SET VANHA=
C:
CD\
```

Kun PATH osoittaa vain BATS-hakemistoon eikä suoraan WP-hakemistoon, WP käynnistyy kuten ennenkin pelkällä komennolla WP, mutta nyt komento ei käynnistäkään itse ohjelmaa vaan komentojonon. Se vaihtaa levyaseman D:ksi, oletushakemiston työhakemistoon ja ottaa käyttöön WP:tä osoittavan polun. Vaikka polkuun periaatteessa riittääkin pelkkä WP:n ohjelmahakemisto, siihen kannattaa lisätä myös

tärkeiden apuohjelmien hakemistot, koska ne näkyvät poistuttaessa ohjelmasta hetkeksi Shelliin eli komentotulkin puolelle.

Kun WP:n käyttö loppuu, komentojonon loppuosa siivoaa jäljet ja palaa takaisin C: aseman päähakemistoon. Tätä periaatetta on helppo laajentaa niin, että paluu tapahtuu yksinkertaiseen valikko-ohjelmaan, jolloin käyttäjän ei tarvitse edes muistaa käytössä olevien ohjelmien nimiä tai niiden käynnistyskomentoja.

Käynnistysjonoa on edelleen helppo laajentaa niin, että sille annetaan parametrina työhakemiston nimi. Komento WP KIRJEET tai WP JUTUT vaihtaa ensin oikeaan hakemistoon ja käynnistää WP:n vasta sitten. Tämä lisäys saadaan muokkaamalla CD-komentoa seuraavasti:

```
CD \WP\%1
```

Käynnistysjonon alkuun voidaan vielä lisätä tarkistus, joka pakottaa käyttäjän antamaan työhakemiston nimen. Samalla tarkistetaan, että hakemistonimi on todella olemassa:

```
IF %1x==x GOTO EIKAY
GOTO TARKISTA
:EIKAY
ECHO Kirjoita WP työnimi, kuten
ECHO WP JUTUT
GOTO ULOS
:TARKISTA
REM tutkitaan, onko annettu hakemisto olemassa
IF EXIST D:\WP\%1\NUL GOTO OK
ECHO nimeä %1 ei ole määritelty
GOTO ULOS
:OK
REM --- vasta tästä alkaa ohjelman käynnistys
D:
CD \WP\TYOT
SET VANHA=%PATH%
PATH=D:\WP;C:\DOS;C:\UTIL;C:\BATS
WP
SET PATH=%VANHA%
SET VANHA=
C:
CD\
:ULOS
```

146

Työtiedostojen automaattinen varmistus

Kun sovellukset käynnistetään komentojonojen avulla, niihin on helppo lisätä automaattinen varmuuskopiointi. Tätä varten levylle perustetaan oma hakemisto (esimerkiksi D:\VARA), johon sovelluksen päätyttyä kopioidaan kaikki muuttuneet tiedostot. Koska yhden istunnon aikana muutetaan yleensä vain muutamia tiedostoja, niiden varmistus tapahtuu hetkessä ja lähes käyttäjän huomaamatta.

Varmistuksessa pitää käyttää XCOPYä ja sen /M-valitsinta, jolloin jokainen muuttunut työtiedosto saadaan mukaan kopiointiin. Esimerkiksi Wordin tapauksessa komentojono voisi näyttää seuraavalta:

```
ECHO OFF
ECHO käynnistän Wordin, odota hetki...
D:
CD \WORD
WORD
ECHO varmistan vielä muuttuneet työtiedostot...
XCOPY *.DOC D:\VARA /M
C:
CD\
ECHO Valmis!
```

Ne sovellukset, jotka eivät käytä työtiedostoilla vakiotarkenninta (kuten WordPerfect), ovat pieni ongelma, koska työtiedostoja ei voi erottaa pelkän tarkentimen perusteella. Silloin joudutaan kopioimaan kaikki hakemiston muuttuneet tiedostot XCOPY *.* -komennolla. Jotta ohjelmatiedostot eivät tulisi mukaan kopiointiin niiden arkistointimäärе pitää poistaa ATTRIB -A -komennolla.

Automaattinen varmistus on sitä luotettavampaa, mitä kauemmaksi kopiotiedosto saadaan alkuperäisestä. Jos mikrossa on vain yksi asematunnus, kaikki kopiointi pitää tehdä saman aseman sisällä, jolloin esimerkiksi kirjanpitoon tullut vika saattaa tuhota niin alkuperäisen tiedoston kuin siitä tehdyn varmistuksenkin.

Silti kopioinnista on hyötyä, vaikka se tapahtuisi yhden aseman sisällä. Jos käyttäjä esimerkiksi vahingossa poistaa tärkeän työtiedoston, sen kopio löytyy yhä toisesta hakemistosta.

Jos kopiointi voidaan tehdä toiselle loogiselle levyasemalle (kuten esimerkin D:), se on turvassa kirjanpitoon tulevilta vioilta mutta ei hyödytä mitään, jos koko levyasema särkyy. Mikäli koneessa on kaksi fyysistä levyasemaa varmistukset kannattaa ohjata toiselle levylle, jolloin ne on turvattu levynkin rikkoutumista vastaan. Jos työtiedostoja käsitellään molemmilla levyillä varmistukset kannattaa ohjata ristiin niin, että molemmilla levyillä olevista tiedostoista on varmistus toisella levyllä.

Tästäkään järjestelystä ei ole apua jos mikro varastetaan tai jos se tuhoutuu vaikkapa tulipalossa. Turvallisinta olisikin kopioida muuttuneet tiedostot verkon kautta kokonaan toiseen koneeseen.

Jotta varmistukset eivät kumuloituessaan täyttäisi koko levyasemaa, ne kannattaa säännöllisin väliajoin poistaa. Komentojonojen yhteydessä on esimerkkejä jonoista, jotka aktivoituvat joko haluttuna päivänä tai joka n:nnellä käynnistyskerralla. Ne sopivat hyvin tyhjentämään \VARA-hakemiston säännöllisin väliajoin.

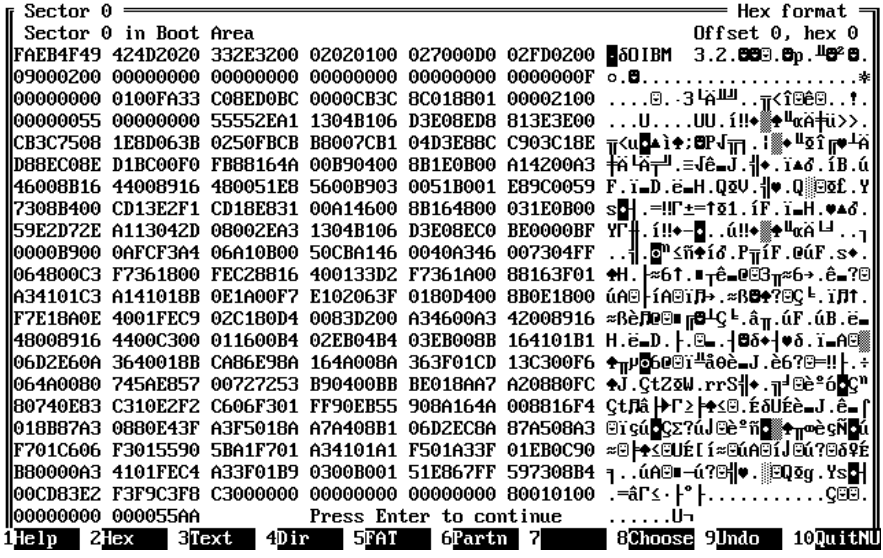
Virustorjunta

147

Käynnistyslohkoviruksen voi nähdä paljaalla silmällä

Levykkeen käynnistyslohko sisältää joukon teknistä tietoa levystä sekä pienen latausohjelman, jonka lopussa on selväkielisiä virheilmoituksia. Kuten kohdassa 73 kävi ilmi, nämä ilmoitukset tulostuvat ruudulle jos käyttöjärjestelmän lataus ei jostain syystä onnistu.

Kun levykkeen käynnistyslohkoon tarttuu virus, se joutuu yleensä kirjoittamaan koko lohkossa olevan latausohjelman uudelleen ja silloin



Levykkeen käynnistyslohkoon (boot sector) pesiytyneen viruksen voi usein nähdä paljaalla silmällä, koska lohkon lopussa olevat selväkieliset virheilmoitukset ovat peittyneet viruksen koodin alle. Kuvassa Disk Killer-viruksen saastuttama käynnistyslohko. Vertaa sivun 121 kuvaan.

selväkieliset ilmoitukset peittyvät viruksen oman koodin alle. Ne tulostuvat kyllä edelleen ruudulle, sillä virus tallentaa alkuperäisen käynnistyslohkon toiseen kohtaan levykettä ja hakee sen sieltä kun käynnistystä yritetään.

Selväkieliset virheilmoitukset on helppo havaita paljaalla silmälläkin, kun käynnistyslohkoa tutkitaan Nortonilla tai PC Toolsilla: jos lohkon lopusta ei löydy virheilmoituksia, on hyvin todennäköistä että levykkeellä on virus. Heksadesimaalisesta koodista ei tarvitse ymmärtää mitään.

Vain muutama virus on tehty niin pieneksi, että ne mahtuvat pelkän latausosan päälle ja lohkon lopussa olevat selväkieliset ilmoitukset jäävät paikoilleen.

148

Tyhjäkin levyke voi sisältää viruksen

Jokaisella alustetulla levykkeellä on käynnistyslohko, vaikka levyke muuten olisi täysin tyhjä. Niinpä myös kaupassa myytävät valmiiksi alustetut levykkeet sisältävät käynnistyslohkon — ja jos huonosti käy myös viruksen. Epäillään, että FORM-virus olisi levinnyt juuri valmiiksi alustettujen tyhjien levykkeiden myötä. Harva mikronkäyttäjä on osannut epäillä, että tyhjäkin levyke voisi olla saastunut.

Jotta vaara minimoituisi, kannattaa varmuuden vuoksi alustaa jokainen ostettu levyke uudelleen. Jos tämä tuntuu liian hitaalta ja työläältä, voi DOS 5:stä lähtien käyttää komentoa

```
FORMAT A: /Q /U
```

Samalla kun FORMAT tyhjentää jo ennestään tyhjän kirjanpidon (/Q -valitsin), se kirjoittaa levykkeen käynnistyslohkon uudelleen, jolloin siinä mahdollisesti ollut virus tuhoutuu. Valitsimet /Q ja /U yhdessä takaavat sen, ettei "alustaminen" kestä epäkäytännöllisen pitkään, kuten pelkällä FORMATilla kävisi.

Toinen tapa hoitaa asia on tarkistaa virusten etsintäohjelmalla kaikki kaupasta ostetut levykkeet — myös tyhjät.

Jos levyke on täysin neitseellinen ja vaatii normaalin alustuksen ennen käyttöä tartuntavaaraa ei luonnollisesti ole.

149

Virukset ja CHKDSK

Monet virukset — varsinkin sellaiset jotka tarttuvat käynnistyslohkoon — varaavat itselleen keskusmuistista kiinteän alueen, joka näkyy perusmuistin kokonaismäärän laskuna. Tavallisesti CHKDSK-listauksen lopussa lukeva kokonaismuisti (total bytes memory) näyttää seuraavalta:

```
655360 total bytes memory
512720 bytes free
```

Seuraavassa taulukossa on lueteltu joitakin yleisiä osiotaulukkoon tai käynnistyslohkoon tarttuvia viruksia, jotka saavat CHKDSK:n näyttämän muistimäärän laskemaan:

Michelangelo	653.312
Music Bug	651.264
Form	651.264
Stoned	651.264
Tequila	652.288
Invader	650.240
Flip	648.192
Joshi	649.216

Tämä tieto on erittäin hyödyllinen kun sitä käytetään oikein päin. Taulukko ei kerro, mikä virus koneessa on, mutta se kertoo, mitä virusta koneessa *ei* ole. Esimerkiksi keväällä 1992 suurta kohua aiheuttanut

Michelangelo-virus sai monet käyttäjät huolestuneina kyselemään, oliko heidän koneessaan virusta. Asia olisi ollut helppo selvittää CHKDSK:n avulla. Jos muistia olisi ollut 655360, koneessa ei olisi ollut ainakaan Michelangeloa.

Joissakin koneissa oleva BIOSin ylimääräinen data-alue varaa itselleen yhden kilotavun muistia, joten perusmuistin määrä on alun pitäenkin vain 639 kiloa eli 654336 tavua. Tämä ero pitää ottaa huomioon myös virusten viemää muistia laskettaessa.

150

FORMAT ei välttämättä tuhoa virusta

Kuten niksissä 66 kävi ilmi, kiintolevyn alustaminen FORMATilla ei todellisuudessa tuhoa levyllä olevia tiedostoja. Tyhjentämällä kirjanpidon FORMAT saa levyn näyttämään tyhjältä, vaikka tiedostoihin ei olekaan koskettu.

Sama pätee myös niihin viruksiin, jotka tarttuvat osiotaulukkoon. Koska alustus ei koske osiotaulukkoon lainkaan, siellä oleva virus selviää helposti alustuksen yli. Viruksen poistaminen edellyttää koko levyn täydellistä tyhjentämistä esimerkiksi Nortonin tai PC Toolsin tyhjennysohjelmien avulla tai suorittamalla levylle perusalustuksen, jos se levyn tekniikan puolesta on mahdollista.

DOS 5.0:sta alkaen on olemassa helpompikin tapa: FDISK /MBR.

151

FDISK /MBR

DOS 5.0:sta lähtien FDISK-ohjelmaan on lisätty dokumentoimaton /MBR-valitsin, josta on suurta apua tuhottaessa osiotaulukkoon tarttunutta viruksia. Komennolla

```
FDISK/MBR
```

ohjelma kirjoittaa osiotaulukossa olevan latausohjelman uudelleen, mutta ei koske taulukossa oleviin numerotietoihin. Jos ohjelmaan on tarttunut virus, se kuolee, mutta taulukko säilyy ennallaan. Mikäli kone on puhdas, komennosta ei ole mitään haittaa.

Dokumentoimaton /MBR-valitsin tarjoaa nopean ja tehokkaan tavan päästä eroon sellaisista viruksista kuten Michelangelo, Joshi ja Stoned. On kuitenkin tärkeätä, että kone käynnistetään puhtaalta levykkeeltä ennen FDISK /MBR-komennon antamista! Jos virus on aktiivisena muistissa kun puhdistuskomento annetaan, virus tarttuu välittömästi takaisin eikä puhdistuksesta ole mitään hyötyä.

FDISK /MBR tappaa kaikki ne virukset, jotka tarttuvat osiotaulukossa olevaan ohjelmaan. Valitettavasti evoluutio on DOS 5.0:n jälkeen tehnyt tehtävänsä ja eräät virukset ovat oppineet selviämään /MBR-valitsimesta.

Venäjältä kotoisin oleva Starship-virus oli ensimmäinen, joka keksi miten tarttua osiotaulukkoon ilman, että siellä olevaan ohjelmaan pitää lainkaan koskea. Ohjelman sijaan Starship muokkaa osiotaulukon tietorivejä niin, että ensimmäinen osio näyttää alkavan viruksen omasta koodista. Näin käynnistystä suorittava DOS ajaa ensiksi viruksen oman koodin ja hyppää vasta sitten varsinaiseen käyttöjärjestelmään — eikä latausohjelmaan ole tarvinnut koskea lainkaan!

FDISK /MBR ei myöskään tehoa FORM-virukseen, joka kiintolevylläkin tarttuu käynnistyslohkoon eikä osiotaulukkoon kuten muut. FORMista pääsee kuitenkin eroon siirtämällä SYSin avulla käyttöjärjestelmän uudelleen. Samalla kun SYS siirtää käyttöjärjestelmän, se kirjoittaa myös käynnistyslohkon uudelleen.

152

Automaattinen pituustarkistus

Monet tiedostovirukset — eivät kuitenkaan kaikki — tarttuvat COMMAND.COMiin, jolloin ohjelmatiedoston pituus kasvaa viruksen pituuden verran. Komentotulkin pituuden kasvu 300-5000 tavulla onkin lähes varma merkki virustartunnasta.

Koska COMMAND.COMin tarkkailu silmämääräisesti on hankalaa, työ kannattaa automatisoida. Komentojono COMTARK.BAT näyttää seuraavalta:

```
ECHO OFF
DIR %COMSPEC% | FIND "47845" >TEMP1.TMP
COPY TEMP1.TMP TEMP2.TMP
IF EXIST TEMP2.TMP GOTO OK
ECHO Komentotulkin pituus on muuttunut!
ECHO Pituuden pitäisi olla 47845 tavua.
ECHO Varmista, ettei koneessasi ole virusta.
:LOOPPI
GOTO LOOPPI
:OK
IF EXIST TEMP2.TMP DEL TEMP2.TMP
DEL TEMP1.TMP
```

Jonoa kutsutaan AUTOEXEC.BATista CALL-komennolla. Aluksi se poimii DIR-listauksesta komentotulkista kertovan rivin. Haku tehdään COMSPEC-ympäristömuuttujan pohjalta, joten tulkki löytyy olipa se sitten pää- tai alihakemistossa.

FIND-komento etsii riviltä komentotulkin oikeata pituutta, joka esimerkin DOS 5.0:ssa on ollut 47845 tavua. Jos lukema löytyy, komentotulkista kertova rivi tulostetaan tiedostoon TEMP1.TMP. Ellei lukua löydy, tiedosto jää tyhjäksi ja sen pituudeksi tulee nolla tavua. Seuraavalla rivillä oleva COPY-käsky ei kopioi nollatavun mittaista tiedostoa, joten TEMP2.TMP jää syntymättä. Tämä on merkki siitä, että komentotulkin pituus on muuttunut joten käyttäjälle tulostetaan varoitus asiasta.

Komentojonon lopuksi poistetaan ensimmäinen aputiedosto ja jos toinen tiedosto on syntynyt, myös se. IF EXIST-lauseella estetään virheilmoitus File not found, ellei toista aputiedostoa ole.

Esimerkin komentojono on nopea ja huomaamaton, joten sen voi lisätä vaikka yrityksen jokaiseen mikroon. Lainausmerkeissä oleva pituustieto pitää kuitenkin tarkistaa konekohtaisesti, joten komentojonoa ei voi suoraan kopioida mikrosta toiseen. Vielä tehokkaammaksi suojan saa, jos samalla periaatteella tarkistetaan yleisesti käytettyjen sovellusohjelmien ja DOSin apuohjelmien pituudet. Esimerkiksi WP:n pituutta tarkistettaessa rivi voisi näyttää tällaiselta:

```
DIR D:\WP\WP.EXE|FIND "266745" >TEMP1.TMP
```

Yksinkertaisuudestaan huolimatta komentojono paljastaa nopeasti ja tehokkaasti tiedostoihin tarttuneet normaalit virukset. Se ei kuitenkaan tehoa ns. stealth-viruksiin, jotka osaavat naamioida itsensä niin, ettei tiedoston pituus näytä muuttuvan vaikka virus onkin kiinnittynyt sen perään.

152 B

Tiedostojen suojaamisesta ATTRIB +R -komennolla ei yleensä ole mitään hyötyä viruksia vastaan. Lähes kaikki virukset osaavat poistaa määreet, tarttua tiedostoon ja laittaa määreet takaisin. Siksi esimerkiksi COMMAND.COMin suojauksesta lukumääreellä ei ole mitään hyötyä viruksia vastaan. Se kannattaa kuitenkin tehdä, sillä kun komentotulkki on kirjoitussuojattu, käyttäjä ei vahingossa pysty poistamaan sitä eikä kopioimaan sen päälle levykkeeltä jonkin toisen DOS-version komentotulkkia.

153

Tiedostojen vertailu FC:llä

Vielä tehokkaampi edellisen niksin ohjelmasta saadaan, mikäli alkuperäinen komentotulkki kopioidaan turvaan ja nimetään uudelleen niin, että sitä voidaan verrata kulloinkin käytössä olevaan tulkkiin. Varmuuden vuoksi alkuperäinen tulkki kannattaa nimetä kokonaan toiseksi, jotta virus ei huomaisi koko ohjelmatiedostoa lainkaan. Seuraavassa esimerkissä on käytetty nimeä KOMMAND.KOM.

Komentojono, joka vertaa käytössä olevaa tulkkia referenssitiedostoon, käyttää MS-DOSin mukana tulevaa FC-vertailuohjelmaa. Se on monipuolisempi kuin PC-DOSissa oleva COMP, joka ei edes aloita vertailua mikäli tiedostojen pituudet poikkeavat toisistaan. FC on kuitenkin sikäli rajoittunut, ettei se palauta tietoa vertailun onnistumisesta tai epäonnistumisesta ERRORLEVELinä vaan tieto on haettava ohjaamalla ohjelman tulostus tiedostoon ja tutkimalla sitä.

Varsinainen vertailukomentojono on esitetty seuraavalla sivulla.

```
ECHO OFF
IF NOT EXIST \VARA\KOMMAND.KOM GOTO VIRHE
FC /B %COMSPEC% \VARA\KOMMAND.KOM |
                    FIND "no differences" >TEMP1.TMP
COPY TEMP1.TMP TEMP2.TMP
IF EXIST TEMP2.TMP GOTO OK
ECHO ^G^G Komentotulkin sisältö on muuttunut!
ECHO Varmista, ettei tiedostossa ole virusta.
:LOOPPI
GOTO LOOPPI
:VIRHE
ECHO Vertailutiedostoa ei löydy!
GOTO LOPPU
:OK
IF EXIST TEMP2.TMP DEL TEMP2.TMP
DEL TEMP1.TMP
:LOPPU
```

Jos mikrossa on vain COMP-ohjelma, komentojonoa pitää muuttaa sillä COMP-ohjelma vastaa eri tavalla kuin FC ja kysyy aina vertailun jälkeen, halutaanko vertailla lisää tiedostoja. Jotta kysymys saataisiin ohitettua, levylle pitää luoda tiedosto N.N, joka sisältää kieltävän vastauksen kysymykseen. Englanninkielisessä DOSissa tiedoston sisällöksi riittää N, rivinvaihto ja Ctrl+Z. Se annetaan COMPille rivin loppuun lisättävällä kulmamerkillä. Lopulliseksi vertailuriviksi saadaan siten

```
COMP %COMSPEC% \VARA\KOMMAND.KOM |
        FIND "compare OK" >TEMP1.TMP <n.n
```

Samalla periaatteella voitaisiin seurata myös käynnistyslohkoa lukemalla se DEBUGin avulla tiedostoksi ja vertailemalla sitä säännöllisesti alkuperäiseen. Käytännössä näin ei kuitenkaan kannata tehdä, sillä kiintolevyn käynnistyslohkoon tarttuvat virukset ovat harvinaisia. Osiotaulukon seuraaminen olisi paljon mielekkäämpää, mutta koska se sijaitsee normaalin levyalueen ulkopuolella, DEBUG ei ulotu sinne.

154

Automaattinen muistimäärän tarkistus

Käynnistyslohkoa mielekkäämpi tarkkailun kohde on perusmuistin määrä. Sitä vähentävät kaikki sellaiset virukset, jotka eivät tartu yksittäisiin ohjelmatiedostoihin vaan kiintolevyn osiotaulukkoon. Muistin määrän tarkistus täydentää siten sopivasti edellä esitettyä tiedostojen pituustarkkailua.

Tarkistuksen suorittava komentojonokin on pitkälti samanlainen. Erona on nyt vain MEM-apuohjelma, jolla muistin kokonaismäärä saadaan nopeasti. Jos DOS-versio on 3.3 tai vanhempi, pitää käyttää CHKDSKiä. Se on kuitenkin hitaampi käyttää ja jos levyn kirjanpidosta löytyy virheitä, komentojono pysähtyy odottamaan käyttäjän kuittausta.

Tarkistus MEMin avulla tehdään seuraavasti:

```
ECHO OFF
MEM|FIND "655360 bytes total conventio"  >TEMP1.TMP
COPY TEMP1.TMP TEMP2.TMP
IF EXIST TEMP2.TMP GOTO OK
ECHO Muistin määrä on laskenut!!
ECHO Tarkista, ettei asialla ole ollut virus.
:LOOPPI
GOTO LOOPPI
:OK
IF EXIST TEMP2.TMP DEL TEMP2.TMP
DEL TEMP1.TMP
```

Jos koneessa on laajennettu BIOS data-alue ja perusmuistia vain 654336 tavua, tämä lukema pitää kirjoittaa komentojonossa olevan 655360:n paikalle.

155

Levykekäynnistyksen estäminen

Kun mikro käynnistetään, se käy tutkimassa onko A: asemassa levykettä. Jos levyke löytyy, mikro yrittää käynnistää itsensä siltä ja jos levykkeen käynnistyslohkossa on virus, se pääsee yrityksen aikana pujahtamaan kiintolevylle. Viimeksi käytetty levyke unohtuu helposti A: asemaan esimerkiksi tiedostojen kopioinnin jälkeen ja tarjoaa näin virukselle tavan pujahtaa kiintolevylle.

Tietoturvan lisäämiseksi uusien PC-mallien BIOSeissa (kuten AMI) ja SETUP-ohjelmissa on yleensä kohta, josta voidaan estää A: aseman tutkiminen käynnistyshetkellä. B: asemasta ei tarvitse huolehtia, sillä vaikka sen valo palaisikin hetken kun mikro käynnistyy, asemaa ei kuitenkaan käytetä eikä sinne unohtuneesta levykkeestä ole mitään vaaraa. Vielä parempi on, jos SETUPista voidaan vaihtaa käynnistysjärjestys niin, että mikro yrittää ensin C:tä ja vasta sen jälkeen A:ta. Näin käyttäjä ei jää pulaan silloinkaan, kun kiintolevy hajoaa eikä käynnistys siltä enää onnistu.

Vaikka A: asemasta käynnistyminen estetäänkin, se ei mitenkään vaikuta A: aseman muuhun käyttöön. Koska todellinen tarve levykekäynnistykseen on pelejä lukuunottamatta erittäin harvinainen eikä käynnistysjärjestyksen muuttamisesta ole mitään haittaakaan, se kannattaa vaihtaa, jos SETUPissa vain suinkin on siihen mahdollisuus.

156

ZIP-pakettien testaus

Jos virustarkistus halutaan ulottaa myös pakattuihin tiedostoihin, paketit pitää purkaa ennen tarkistusta. Mikäli paketteja on paljon, tarkistus kannattaa kirjoittaa komentojonoksi, joka käy läpi kaikki paketit, purkaa ne yksi kerrallaan, tarkistaa paketista purkautuneet ohjelmat sekä antaa varoituksen, mikäli niistä löytyy viruksia.

Tarkistus vaatii kaksi erillistä komentojonoa. Tarkistusta ajavana jonona toimii TARKZIP.BAT, joka näyttää seuraavalta:

```
ECHO OFF
BREAK ON
FOR %%I IN (*.ZIP) DO CALL SCANZIP %%I
```

Se ottaa kaikki oletushakemistossa olevat ZIP-tiedostot ja välittää ne yksi kerrallaan varsinaiselle tutkimisohjelmalle, SCANZIP.BATille. Break kytketään päälle, jotta komentojonon keskeyttäminen myöhemmin olisi helpompaa sen jäädessä luuppiin viruksen löydyttyä.

Tutkimisohjelma kopioi saamansa ZIP-tiedoston TEMP-nimiseen apuhakemistoon, avaa paketin siellä ja suorittaa virustarkistuksen. Keskusmuistia ei kannata tarkistaa, koska vaikka paketissa olisikin virus, se ei voi aktivoitua koska sitä ei ajeta. Muistin tarkistaminen jokaisen ZIP-tiedoston kohdalla vain hidastaisi komentojonon läpimenoaikaa.

Jos tarkistusohjelma palauttaa ERRORLEVEL-arvon nolla, kaikki on kunnossa ja puretut tiedostot saa poistaa saman tien. Jos arvo poikkeaa nollasta, jotain on sattunut. Ohjelmassa on joko virus tai etsintäohjelman ajossa on sattunut jokin virhe. Tällöin annetaan äänimerkki Ctrl+G:llä ja pysäytetään komentojono.

```
COPY %1 TEMP
CD TEMP
PKUNZIP %1
IF ERRORLEVEL 1 GOTO VIRHE
SCAN *.* /NOMEM
IF ERRORLEVEL 1 GOTO LOYTYI
GOTO LOPPU
:LOYTYI
ECHO ^G^GPaketissa %1 on mahdollisesti virus!
:LUUPPI
GOTO LUUPPI
:VIRHE
ECHO paketin purkaminen keskeytyi virheeseen,
ECHO paketti voi olla viallinen tai levy täynnä
GOTO LUUPPI
:LOPPU
ECHO Y | DEL *.*
CD ..
```

Jos tutkitaan muita kuin ZIP-paketteja tai jos tarkistusohjelma on jokin muu kuin McAfeen SCAN komentojonoon pitää tehdä pieniä muutoksia, mutta periaate säilyy samana. Samaa periaatetta voi soveltaa myös silloin, kun suureen joukkoon ZIP-paketteja halutaan tehdä jotakin muutoksia tai kun paketeista halutaan etsiä ehdot täyttäviä tiedostoja tms.

157

Miten erottaa väärä hälytys oikeasta?

Virusten nopea määrällinen kasvu johtaa väistämättä väärien hälytysten lisääntymiseen. Etsintäohjelmat joutuvat käyttämään yhä pidempiä sormenjälkiä erotellakseen virukset luotettavasti toisistaan ja estääkseen niitä aiheuttamasta vääriä hälytyksiä. Myös monimutkaiset, itseään koodaavat ja muuntelevat supervirukset vaikeuttavat tilannetta ja tuottavat helposti vääriä hälytyksiä. Ja hyvä niin, sillä parempi väärä hälytys silloin tällöin kuin ei hälytystä silloin, kun sitä todella tarvittaisiin.

Väärien hälytysten kanssa on vain opittava elämään. Siksi on hyvä tietää merkkejä, joiden perusteella voi arvioida onko hälytys aito vai väärä.

Hälytys on todennäköisesti väärä

- jos se tulee työtiedostosta
- jos se tulee vain yhdestä kiintolevyn ohjelmatiedostosta, vaikka tiedetään että ohjelmaa on ajettu
- jos vain yksi virustenetsintäohjelma antaa sen
- jos se tulee muistista silloin, kun on juuri käytetty jotain toista etsintäohjelmaa

Hälytys on todennäköisesti oikea, jos se tulee

- muistista, kun muita etsintäohjelmia ei ole käytetty
- osiotaulukosta tai käynnistyslohkosta

- useasta levyllä olevasta ohjelmasta
- useammalla kuin yhdellä etsintäohjelmalla

Virukset voivat tarttua vain ohjelmatiedostoihin tai levyn varattuihin alueisiin. Ne eivät siten voi tarttua työtiedostoihin ja vaikka tarttuisivatkin, niistä ei olisi mitään vahinkoa, koska työtiedostossa olevaa bittimassaa ei koskaan ajeta ohjelmana. Siksi muiden kuin COM- ja EXE-loppuisten tiedostojen tutkiminen on yleensä ajanhukkaa ja tuottaa helposti vääriä hälytyksiä.

Eräät ohjelmat on kuitenkin tehty niin, että osa niiden koodista sijaitsee levyllä tiedostossa, jonka tarkennin ei ole COM eikä EXE. Kun tällaista ohjelmaa ajetaan viruksen ollessa muistissa, se tarttuu ohjelman tavoin käynnistyvään tiedostoon sen tarkentimesta riippumatta. Siten muiden kuin selvästi COMeiksi tai EXEiksi merkittyjen tiedostojen tutkimista ei voi pitää aivan turhanakaan. Esimerkiksi MNU-tiedostot sisältävät joissakin ohjelmissa ajettavaa koodia ja ne on syytä tarkistaa. Myös PIF-tiedostoja kohdellaan ohjelmien tavoin ja siksi nekin on syytä käydä läpi. PIF-tiedoston pituuden pitäisi olla joko 369 tavua (vanhat PIFit) tai 545 tavua (Windows 3.0:sta alkaen). Jos pituus poikkeaa näistä, asialla voi olla virus.

On kuitenkin selvää, että jos varoitus saadaan tiedostosta, joka on selvästi työtiedosto (kuten BMP, PCX, DOC, XLS jne) ja joka toimii sovelluksella avattaessa normaalisti, kyseessä on väärä hälytys.

Myös tiedostojen pysyvyyttä tarkkailevat ohjelmat saattavat antaa vääriä hälytyksiä. Ne laskevat tiedostoista tarkistussummia, jolloin viruksen tarttuminen ohjelmaan aiheuttaa summan muuttumisen ja siten hälytyksen. Valitettavasti jotkut ohjelmat, kuten DOSin oma SETVER, muuttavat eräissä tilanteissa itse itseään ja saattavat näin laukaista aiheettoman hälytyksen.

Eräät tarkistussummiin pohjautuvat ohjelmat on tehty niin suoraviivaisesti, että ne tarkistavat kaikki SYS-loppuiset tiedostot — myös CONFIG.SYSin. Hälytys saadaan aina, kun tiedostoa muokataan.

158

Bad sectorien määrä ei voi kasvaa itsestään

DOSissa on vain kaksi ohjelmaa, jotka voivat kasvattaa kirjanpidossa olevien Bad sector-merkintöjen määrää — eikä kumpikaan niistä ole CHKDSK, joka ainoastaan raportoi kirjanpitoalueilla olevat merkinnät käyttäjälle. Se ei koskaan itse lisää Bad sectorien määrää. Se ei liioin lue levyä alusta loppuun ja tutki, onko sille tullut uusia vika-alueita, jotka pitäisi merkitä.

Vika-alueiden kartoituksen ja merkinnän kirjanpitoon suorittaa ainoastaan yksi ohjelma: FORMAT. Jos FORMATia ei ole käytetty, levylle merkittyjen Bad sector -alueiden koon tai määrän ei pitäisi muuttua. Jos ne muuttuvat, asialla voi olla virus, joka kopioi itsensä levylle ja naamioi alueen sen jälkeen Bad sector -merkinnällä jotta DOS ei kirjoittaisi sen päälle. Varsinkin levykkeillä Bad sectoreihin piiloutuminen on yleistä. Kiintolevyillä sitä harrastavat vain muutamat virukset, koska piilopaikkoja löytyy muutenkin.

Eräät virukset — kuten FORM — kirjoittavat itsensä levyn vapaan alueen loppuun, mutta eivät merkitse aluetta käytetyksi kirjanpitoon. Virukset luottavat siihen, että levy ei koskaan tule aivan täyteen. Jos levy täyttyy, viruksen koodi tuhoutuu ja virus kuolee.

Toinen Bad sector-merkintöjen lisäämiseen pystyvä ohjelma on RECOVER. Kun sitä käytetään vioittuneen tiedoston pelastamiseen, se merkitsee työn jälkeen vikakohdan kirjanpitoon niin, että aluetta osataan jatkossa välttää.

Näiden DOSin ohjelmien ohella on lukuisia kaupallisia apuohjelmia, jotka pystyvät lukemaan levyn alusta loppuun ja jos uusia vika-alueita löytyy, ne merkitsevät viat kirjanpitoon. Näiden ohjelmien käyttö on ainoa mahdollisuus silloin, jos levyyn on tullut vikaa eikä sitä haluta tiedostojen menettämisen vuoksi alustaa FORMATilla.

159

Virusetsinnän seuranta verkossa

Lähiverkko helpottaa virusten leviämistä, jos verkon hallinta on muuten retuperällä eikä oikeuksien jaosta ole huolehdittu kunnolla. Jos niistä on huolehdittu, verkosta on pelkkää etua virustorjunnassa. Verkko mahdollistaa keskitetyt ohjelmapäivitykset niin, että jokaisella käyttäjällä on aina uusin versio etsintäohjelmasta. Se mahdollistaa myös keskitetyn seurannan, jossa tukihenkilö voi omalta koneeltaan tarkistaa työasemissa viimeksi ajettujen etsintäohjelmien raportit.

Keskitettyä tarkkailua varten luodaan jokaiseen työasemaan komentojono, joka ajaa etsintäohjelman ja lähettää sen ajossa syntyneen raportin tiedostoon. Työasemassa ilmoitetaan ensin mikron nimi sijoittamalla se ympäristömuuttujaan

```
SET USER=Jaakko
```

ja varsinainen tarkistus tehdään seuraavalla komentojonolla:

```
ECHO OFF
ECHO Virustarkistus meneillään, odota hetki...
REM Poistetaan vanhin raportti
DEL Y:%USER%.BA2
REM tehdään edellisistä ajoista varmuuskopiot
COPY Y:%USER%.BA1 Y:%USER%.BA2
COPY Y:%USER%.SCN Y:%USER%.BA1
REM varsinainen tarkistus alkaa
SCAN C: /REPORT Y:%USER%.SCN
IF ERRORLEVEL 1 GOTO ONVIRUS
ECHO Tarkistus OK!
GOTO LOPPU
:ONVIRUS
REM Haetaan virusraportti myös omaan työasemaan
COPY Y:%USER%.SCN C:\
ECHO Koneessasi saattaa olla virus, ota
ECHO yhteys tukihenkilöön puh. 1234
REM ...ja tulostetaan raportti ruudulle luettavaksi
TYPE C:\%USER%.SCN
CTTY COM1
:LOPPU
```

Jonossa viitataan yhteiseen Y: asemaan, johon jokaisella työasemalla on oltava kirjoitusoikeudet, koska etsintäohjelman tuottama raportti tallennetaan sinne. Raportti syntyy tarkentimelle SCN ja tiedoston nimenä on työaseman käyttäjätunnus, joka saa olla enintään kahdeksan merkkiä. Sen on myös oltava yksilöllinen jotta eri työasemien lähettämät raportit eivät menisi päällekkäin.

Kun tukihenkilö haluaa seurata virusten etsinnän sujumista, hän avaa yhteyden Y: asemalle hakemistoon, jossa raportit ovat. Komennolla

```
DIR /ON
```

hän näkee kaikkien työasemien kolme viimeksi suoritettua ajokertaa. Niiden päiväyksistä on helppo havaita, onko tarkistusta ajettu säännöllisesti. Koska tarkistus kestää oman aikansa, sitä ei yleensä voi laittaa AUTOEXEC.BATiin vaan käynnistys on jätettävä käyttäjän omalle vastuulle.

Virusten torjunta ei ole enää tekninen ongelma, sillä riittävän hyviä ja luotettavia etsintäohjelmia on useita. Jäljelle jääneet ongelmat ovat psykologisia: miten saada käyttäjät muistamaan levykkeiden käsittelystä annetut ohjeet ja todella noudattamaan niitä? Miten taata, että käyttäjät todella ajavat aikaa vievän tarkistusohjelman säännöllisesti? Tämä komentojono helpottaa tilanteen seuraamista.

Tukihenkilö voi myös tehdä omaan käyttöönsä komentojonon, joka käy tarkistamassa kaikki yksittäiset raportit ja näyttää niiden yhteenvedon yhtenä raporttina. Tämä tapahtuu ajamalla tukihenkilön omassa koneessa komentojono TARKISTA.BAT:

```
ECHO OFF
Y:
REM tuhotaan vanha yhteenvetotiedosto
IF EXIST TOTAL.RPT DEL TOTAL.RPT
ECHO tarkistan käyttäjien raportteja, odota...
FOR %%i IN (*.SCN) DO FIND "Found the" %%i
                                          >>TOTAL.RPT
TYPE TOTAL.RPT | MORE
```

Ruudulle tulostuva raportti sisältää työasemien nimet ja jos viruksia on löytynyt, myös niiden nimet:

```
---------- JAAKKO.SCN
    Found the Crew-2480 [2480] Virus

---------- PETTERIJ.SCN

---------- ANNE-S.SCN
  Found the FamM [FM] Virus
```

Matkamikron käyttö

160

Nimikoi koneesi

Matkamikrojen kehittyminen on lisännyt niitä kohtaan tunnettua kiinnostusta paitsi ostajien myös varkaiden keskuudessa. Mitä pienemmäksi ja kevyemmäksi mikrot käyvät, sitä helpompia ne ovat varastaa. Eikä asialla välttämättä tarvitse olla varas: kevyt ja pieni laite on helppo unohtaa junaan tai lentokoneeseen.

Koska myös pöytämikrojen varastaminen on yleistynyt, jokaisen mikronomistajan kannattaa ikuistaa nimensä ja yhteystietonsa mikron sisälle. Paras paikka tietojen tallentamiselle on osiotaulukon jatkona oleva nollaura, joka on lähes kaikissa koneissa tyhjä. Edes FORMAT-ohjelma ei yllä sinne. Levyn sektorimäärästä riippuen tilaa on aina vähintään kahdeksan kilotavua mutta joskus jopa useita kymmeniä kiloja. Tilan puolesta nollauralle voi kirjoittaa pitkätkin selitykset koneestaan.

Jotkut harvat järjestelmät saattavat käyttää nollauraa tai sen alkua omiin tarkoituksiinsa. Samoin tekevät osiotaulukkoon tarttuvat virukset, jotka siirtävät alkuperäisen taulukon toiseen kohtaan nollauraa ja valtaavat itse sen paikan. Jos siis nollauralta löytyy vaihtelevan näköistä koodia, on parasta jättää se rauhaan. Mikäli nollaura on täynnä nollaa tai jotain vakioarvoa, sen päälle voi huoletta kirjoittaa.

Nollauran lukemisen ja kirjoittamisen voi tehdä Nortonin NU tai Diskedit-apuohjelmalla, tai jollain muulla vastaavalla työkalulla. Jos käytetään PC Toolsia, sen version pitää olla 8.0 tai uudempi, koska vanhemmat versiot eivät pysty lukemaan osiotaulukkoa eivätkä nollauraa. Uralle kannattaa kirjoittaa laitteen oikean omistajan nimi ja yhteystiedot. Paras paikka kirjoitukselle on nollauran viimeinen lohko (sektori), koska edes virukset eivät yleensä ulotu sinne.

```
Side 0, Cylinder 0, Sector 17 ══════════════════════════════════ Hex format
                                                         Offset 336, hex 150
20202054 8E524B45 8E548E20 548E524B 458E548E 20202020    TÄRKEÄTÄ TÄRKEÄTÄ
2020203D 3D3D3D3D 3D3D3D3D 3D3D3D3D 3D3D3D3D 20202020    =================
54846D84 206D696B 726F206F 6E207661 72617374 65747475 Tämä mikro on varastettu
20202020 20202020 20202020 20202020 20202020 20202020
4F746120 79687465 7973206D 696B726F 6E206F6D 69732D20 Ota yhteys mikron omis-
74616A61 616E3A20 20202020 20202020 20202020 20202020 tajaan:
50657474 65726920 4A847276 696E656E 20202020 20202020 Petteri Järvinen
5075682E 20283930 29203132 30353731 31202020 20202020 Puh. (90) 1205711
4661782E 20283930 29203436 36343634 20202020 20202020 Fax. (90) 466464
20202020 20202020 20202020 20202020 20202020 20202020
74616920 6C846869 6D708484 6E20706F 6C696973 69696E2E tai lähimpään poliisiin.
20202020 20202020 20202020 20202020 20202020 20202020
4C947974 9470616C 6B6B696F 20202020 20202020 20202020 Löytöpalkkio
20202020 20202020 20202020 20202020 20202020 20202020
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 E5E5E5E5 σσσσσσσσσσσσσσσσσσσσσσσσ
E5E5E5E5 E5E5E5E5          Press Enter to continue    σσσσσσσσ
1Help   2Hex   3Text   4Dir   5FAT   6Partn   7   8Choose 9Undo   10QuitNU
```

Kiintolevyn nollauran viimeiseen sektoriin kirjoitettu teksti on kuin koiran panta: se sisältää omistajan nimen ja yhteystiedot. Tämä auttaa palauttamaan mikron sen oikealle omistajalle.

Jotta niksistä olisi hyötyä, sen täytyy toimia myös toisinpäin: jos sinulle tarjotaan edullisesti käytettyä — tai uuttakin — mikroa, tarkista sen nollauran viimeinen lohko ennen kauppoihin ryhtymistä. Näin varmistat, ettet tule hankkineeksi varastettua tavaraa. Jos tällainen käytäntö saadaan yleistymään, se tulee helpottamaan varastettujen mikrojen palauttamista takaisin niiden oikeille omistajille.

Koska nimikointi on tehty kiintolevylle eikä itse mikrolle, yhteystiedoista ei ole paljoakaan hyötyä jos varas siirtää kiintolevyaseman koneesta toiseen. Koska matkamikrojen levyasemia ei yleensä voi siirtää koneesta toiseen, niksi toimii erityisen tehokkaasti juuri niillä. Mikään ei silti estä nimikoimasta myös pöytäkoneita.

161

Akkujen käyttöiän lisääminen

Matkamikron akut eivät koskaan tunnu kestävän tarpeeksi pitkään. Jotta akuista saisi irti mahdollisimman pitkän käyttöajan kannattaa opetella käyttämään niitä oikein.

Matkamikroissa käytetyt akut ovat NiCD- tai NiMH-tyyppisiä. Nikkelikadmium-akut ovat yleisempiä ja halvempia ja niitä käytetään myös monissa viihde-elektroniikan laitteissa kuten videokameroissa, NMT-puhelimissa ja ladattavissa CD-soittimissa. Ne ovat myös herkempiä väärälle käsittelylle kuin NiMH-akut.

Miten akkuja sitten pitäisi käsitellä, jotta ne kestäisivät mahdollisimman pitkään? Tärkein sääntö on, että akkuja ei saa koskaan ladata jos ne eivät ole tyhjiä. Puolityhjien akkujen lataus tuottaa nopeasti "muistiilmiön", jossa akut pystyvät latautumaan vain edellisillä kerroilla käytettyyn kapasiteettiin asti.

Erityisen kriittinen on ensimmäinen latauskerta. Neitseellinen, paketista otettu akku vaatii vähintään vuorokauden mittaisen yhtäjaksoisen lataamisen ennen kuin se voidaan ottaa käyttöön. Liian lyhyt ensilataus pilaa akun niin, ettei se myöhemminkään pysty latautumaan täyteen.

Jos akussa on jäljellä varausta se pitäisi aina purkaa ennen uutta latausta. Tätä varten voi tehdä komentojonon UUVUTA.BAT, joka ikuisessa silmukassa kuormittaa koko ajan kiintolevyä. Myös taustavalo kannattaa asettaa kirkkaaksi, jotta jäljellä oleva varaus kuluisi nopeammin. Välimuisti pitää poistaa käytöstä jotta kiintolevy todella pyörisi kaiken aikaa. Lopullinen jono UUVUTA voisi näyttää vaikkapa seuraavalta:

```
ECHO OFF
SMARTDRV C-
ECHO Akkujen uuvutus käynnissä, älä koske...
:ALKU
CHKDSK >NUL
TREE  >NUL
DIR /S >NUL
GOTO ALKU
```

Kun tämä komentojono jätetään pyörimään, se imee viimeisetkin mehut akusta. Uusi lataaminen aloitetaan vasta sen jälkeen. Esimerkissä oleva DIR /S edellyttää DOS 5.0:aa tai uudempaa. Jos oma DOS-versio on vanhempi, sen tilalle voi kirjoittaa jotain muita levyä kuormittavia komentoja.

Kun akku on sitten saatu tyhjäksi, se pitää ladata taas täyteen mahdollisimman nopeasti sillä tyhjänä säilyttäminen pilaa akun lopullisesti. Jos akkua joudutaan säilyttämään tai mikroa ei käytetä pitkään aikaan, akku pitää ensin ladata täyteen. Täyteen ladattunakin se purkautuu hitaasti itsekseen, joten pitkän seisokin jälkeen on syytä ajaa muutamaan kertaan lataus/purku ennen kuin mikron kanssa lähdetään esimerkiksi matkalle.

Oikein käsiteltynä akut kestävät useita satoja latauskertoja ilman, että niiden varauskyky laskee enempää kuin 10-20 prosenttia uuteen akkuun verrattuna. Tästä huolimatta akut ovat kulutustavaraa ja niitä joutuu aina välillä uusimaan. Koska vanhat akut ovat ongelmajätettä niitä ei saa viedä kaatopaikalle. Akut on palautettava esimerkiksi myyjälle tai mikron maahantuojalle, joka huolehtii niiden toimittamisesta asianmukaiseen kierrätykseen.

162

Kiintolevyn sammutusajan säätäminen

Akun kestoon voidaan vaikuttaa paitsi akkujen oikealla käsittelyllä myös mikron virransäästöominaisuuksilla. Niistä tärkein on kiintolevylle asetettu sammutusaika. Kun viimeisestä levyoperaatiosta on kulunut haluttu aika, säästöautomatiikka pysäyttää kiintolevyn pyörimisen. Koska levyn osuus virrankulutuksesta on 20-40 prosenttia, sen pysäyttäminen lisää selvästi mikron toiminta-aikaa.

Tarpeettomasti pyörivä levy kuluttaa turhaan virtaa, mutta liian lyhyt sammutusaika on sekin huono ratkaisu. Käynnistyessään levy aiheuttaa nimittäin virrankulutukseen voimakkaan piikin, joten turhat käynnistykset kuluttavat enemmän virtaa kuin tarpeeton pyöriminen. Sammutuksen optimiarvo riippuukin käytettävistä ohjelmista ja työskentelytavoista.

Monissa matkamikroissa akkujen loppuminen voi tapahtua niin yllättäen, että käyttäjällä ei ole aikaa lopettaa työtään ja tallentaa tiedostoja levylle. Siksi automaattista välitallennusta kannattaa käyttää, jos ohjelmassa vain on siihen mahdollisuus. Tallennusajan väli on kuitenkin määrättävä tarkkaan. Pahinta on, jos tallennus tapahtuu esimerkiksi viiden minuutin välein ja sammutusajaksi on määrätty neljä minuuttia. Silloin välitallennus käynnistää levyn ja aiheuttaa virtapiikin pian sen jälkeen kun säästöautomatiikka on sammuttanut levyn pyörimisen. Jopa yhtäjaksoinen pyöriminen kuluttaisi vähemmän sähköä kuin toistuva uudelleenkäynnistys.

Työskentelytapaan voi kiinnittää huomiota niin, että kun levy on välitallennuksen tai jonkin muun syyn vuoksi lähtenyt pyörimään, tehdään saman tien muutkin levyä kuormittavat operaatiot.

163

Välimuisti ja RAM-levy matkamikrossa

Levyn pyörimistä voi vähentää myös ohjelmallisilla keinoilla: välimuistilla ja RAM-levyllä.

Mitä isompi välimuisti, sitä useammat lukupyynnöt voidaan tyydyttää sen avulla ja sitä harvemmin levy joutuu käynnistymään. Tätä voi käyttää hyödyksi myös seuraavasti: kun tiedetään, että käytettävä ohjelma tai työtiedosto on iso, kopioidaan se kerran läpi komennolla

```
COPY ISO.DOC /B NUL
```

ennen työskentelyn aloittamista. Kopioinnin aikana työtiedosto jää välimuistiin, josta se löytyy kun sovelluksen käyttö aloitetaan. Muuten voi käydä niin, että ohjelma lukee tiedostoa pieninä palasina, joista jokainen pakottaa levyn käynnistymään uudelleen. Myös monet ohjelmat ja varsinkin Windows-sovellukset hyötyvät tästä tekniikasta, sillä ohjelman koodista luetaan keskusmuistiin vain pieni osa kerrallaan.

Vielä enemmän on hyötyä RAM-levystä. Kun koko ohjelmatiedosto kopioidaan sille ennen työskentelyn aloittamista ei kiintolevyä tarvitse käynnistää kuin työtiedostojen lukemista ja tallennusta varten. Työtiedostojen säilyttäminen RAM-levyllä ei yleensä ole suositeltavaa, sillä akkujen äkillinen loppuminen saattaa tyhjentää RAM-levyn ja johtaa töiden katoamiseen.

164

Iso kohdistin

Useimmissa mikroissa DOSin merkkipohjainen vilkkuva kohdistinmerkki näkyy alaviivana. Se sopii hyvin pöytäkoneiden näytöille, mutta erottuu huonosti matkamikron LCD-näytöllä.

Ohjelma ISOKURSO.COM saa kohdistimen näkymään isona vilkkuvana laatikkona:

```
N ISOKURSO.COM
E 0100 BB 85 00 8B F3 BB 40 00
E 0108 8E C3 26 8B 04 2D 01 00
E 0110 8B C8 B4 01 CD 10 B8 00
E 0118 4C CD 21
RCX
001B
W
Q
```

Ohjelma kannattaa lisätä AUTOEXEC.BATiin.