Ελληνικό Large Disk HOWTO

Andries Brouwer, aeb@cwi.nl

v2.2m, 15 Φεβρουαρίου 2000

Όλα τα σχετικά με τη γεωμετρία του δίσκου και το όριο των 1024 κυλίνδρων. — Για οποιαδήποτε σχόλια, διορθώσεις, κλπ σχετικά με την ελληνική μετάφραση επικοινωνήστε με τον Παναγιώτη Βουδούρη στη διεύθυνση: panos@panos.uklinux.net

1 Το πρόβλημα

Ας πούμε ότι έχετε ένα δίσκο με 1024 κυλίνδρους. Ας υποθέσουμε ότι έχετε κι ένα λειτουργικό που χρησιμοποιεί την παλιά INT13 BIOS διασύνδεση με το I/O του δίσκου. Τότε έχετε πρόβλημα, αφού αυτή η διασύνδεση χρησιμοποιεί πεδία με 10-bit για τους κυλίνδρους για το I/O, οπότε κύλινδροι πέρα των 1024 δεν είναι προσβάσιμοι.

Ευτυχώς, το Linux δε χρησιμοποιεί το BIOS, οπότε δεν υπάρχει πρόβλημα.

Δηλαδή, εκτός από δυο πράγματα:

(1) Όταν ξεκινά το σύστημα, το Linux δεν τρέχει ακόμα οπότε δεν μπορείτε να αποφύγετε τα προβλήματα του BIOS. Αυτό έχει μερικές επιπτώσεις στο LILO και παρόμοιους φορτωτές εκκίνησης.

(2) Είναι απαραίτητο για όλα τα λειτουργικά που χρησιμοποιούν ένα δίσκο να συμφωνούν που βρίσκονται οι κατατμήσεις. Με άλλα λόγια, εάν χρησιμοποιείτε Linux και, ας πούμε, DOS σε ένα δίσκο, τότε πρέπει και τα δυο να μεταφράζουν τον πίνακα κατατμήσεων με τον ίδιο τρόπο. Αυτό έχει μερικές επιπτώσεις για τον πυρήνα του Linux και το `fdisk`.

Παρακάτω ακολουθεί μια λεπτομερής περιγραφή όλων των σχετικών θεμάτων. Σημειώστε ότι χρησιμοποίησα τον πυρήνα 2.0.8 ως αναφορά. Άλλες εκδόσεις μπορεί να διαφέρουν λιγάκι.

2 Περίληψη

Έχετε έναν καινούριο μεγάλο δίσκο. Τι κάνετε; Από την πλευρά του λογισμικού: χρησιμοποιείτε το `fdisk` (ή, καλύτερα, `cfdisk`) για να δημιουργήσετε τις κατατμήσεις, και μετά το `mke2fs` για να δημιουργήσετε ένα σύστημα αρχείων, και έπειτα `mount` για να προσαρτήσετε το καινούριο σύστημα στην ιεραρχία.

Ένα χρόνο πριν θα μπορούσα να γράψω: Δε χρειάζεται να διαβάσετε αυτό το HOWTO εφόσον *δεν υπάρχουν* προβλήματα με μεγάλους δίσκους αυτές τις μέρες. Η πλειοψηφία των προβλημάτων δημιουργούνται από χρήστες που νομίζουν ότι μπορεί να έχουν πρόβλημα και εγκαθιστούν ένα διαχειριστή δίσκου ή χρησιμοποιούν το `fdisk` σε expert mode, ή δηλώνουν τη γεωμετρία του δίσκου στο LILO ή στον πυρήνα.

Παρόλα αυτά, τυπικά προβλήματα είναι: (α) απαρχαιωμένος εξοπλισμός, (β) πολλά λειτουργικά στον ίδιο δίσκο και, μερικές φορές, (γ) η εκκίνηση.

Αυτές τις μέρες η κατάσταση είναι χειρότερη. Ίσως ο 2.3.21 και νεότεροι πυρήνες να είναι καλοί για όλους του δίσκους.

Συμβουλή:

Για μεγάλους δίσκους SCSI: το Linux τους υποστηρίζει από πολύ νωρίς. Δεν χρειάζεται να κάνετε τίποτα.

Για μεγάλους IDE δίσκους (πάνω από 8.4GB): βρείτε μια νέα σταθερή έκδοση πυρήνα (2.0.34 ή νεώτερη). Συνήθως όλα θα διορθωθούν, ειδικά αν δε ζητήσετε από το BIOS μεταφράσεις LBA και τα σχετικά.

Για πολύ μεγάλους IDE δίσκους (πάνω από 33.8 GB): δείτε το 12.1 (IDE προβλήματα με 34+ GB δίσκους) παρακάτω.

Εάν το LILO κρεμάει κατά την εκκίνηση, δηλώστε και στο `linear` αρχείο ρυθμίσεων `etc/lilo.conf`

Μπορεί να υπάρξουν προβλήματα με τη γεωμετρία που μπορούν να λυθούν με τη δήλωση στον πυρήνα /LILO/fdisk.

Εάν έχετε παλιό `fdisk` και προειδοποιεί για 6 (επικαλυπτόμενες) κατατμήσεις: αγνοήστε το, ή ελέγξτε χρησιμοποιώντας το `cfdisk` ότι όλα είναι πράγματι εντάξει.

Εάν νομίζετε ότι κάτι πάει λάθος με το μέγεθος του δίσκου, σιγουρευτείτε ότι δε μπερδεύετε τις δυαδικές με τις δεκαδικές 3 (), και μάθετε ότι ο ελεύθερος χώρος που αναφέρει το `df` σε έναν άδειο δίσκο είναι λίγο τοις εκατό μικρότερο από το μέγεθος της κατάτμησης αφού υπάρχουν κρατήσεις από το σύστημα.

Αν ακόμα νομίζετε ότι υπάρχει πρόβλημα, ή απλά είστε περίεργοι, διαβάστε παρακάτω.

3 Μονάδες και Μεγέθη

Ένα kilobyte (kB) είναι 1000 bytes. Ένα megabyte (MB) είναι 1000 kB. Ένα gigabyte (GB) είναι 1000 MB. Ένα terabyte (TB) είναι 1000 GB. Αυτό είναι το *SI σύστημα*. Παρόλα αυτά, υπάρχει κόσμος που χρησιμοποιεί το 1 MB=1024000 bytes και μιλά για 1.44 MB δισκέτες και κάποιοι που νομίζουν ότι 1 MB=1048576 bytes. Εδώ ακολουθώ το *προτεινόμενο standard* και γράφω Ki, Mi, Gi, Ti για τις δυαδικές μονάδες, ώστε οι δισκέτες είναι 1440 KiB (1.47 MB, 1.41 MiB), 1 MiB είναι 1048576 bytes (1.05 MB), 1 GiB είναι 1073741824 bytes (1.07 GB) και 1 TiB είναι 1099511627776 bytes (1.1 TB).

Σωστά, οι κατασκευαστές δίσκων ακολουθούν το SI σύστημα και χρησιμοποιούν δεκαδικές μονάδες. Παρόλα αυτά, μερικά μυνήματα του Linux και μερικά fdisk προγράμματα χρησιμοποιούν τα σύμβολα MB και GB για δυαδικά, ή και μικτές δυαδικές-δεκαδικές μονάδες. Για αυτό, πριν νομίσετε ότι ο δίσκος σας είναι μικρότερος από αυτόν που σας υποσχέθηκαν, υπολογίστε πρώτα το πραγματικό του μέγεθος σε δεκαδικές μονάδες (ή απλά bytes).

Σχετικά με την ορολογία και τις συντομεύσεις των δυαδικών μονάδων, ο *Knuth* έχει μια εναλλακτική *πρόταση*, συγκεκριμένα να χρησιμοποιούμε KKB, MMB, GGB, TTB, PPB, EEB, ZZB, YYB και να τα καλούμε *μεγάλο kilobyte*, *μεγάλο megabyte*, ... *μεγάλο yottabyte*. Γράφει 'Σημειώστε ότι το να γράφουμε δυο φορές το κάθε γράμμα υπονοεί και δυαδικό και μεγάλο. Είναι μια καλή πρόταση, αφού το 'μεγάλο gigabyte' ακούγεται καλύτερο από το 'gibibyte'. Για την περίπτωσή μας το μόνο που πρέπει να προσέξουμε είναι ότι ένα megabyte έχει ακριβώς 1000000 bytes, και κάποιος άλλος όρος και συντόμευση χρειάζεται αν εννοούμε κάτι άλλο.

3.1 Μέγεθος τομέα

Στο παρόν κείμενο ένα τομέας έχει 512 bytes. Αυτό είναι σχεδόν πάντα αλήθεια, εκτός από μερικούς ΜΟ που χρησιμοποιούν 2048 bytes, και όλες οι χωρητικότητες που δίνονται παρακάτω πρέπει να πολλαπλασιαστούν επί τέσσερα. (Όταν χρησιμοποιείτε το `fdisk` σε τέτοιους δίσκους , σιγουρευτείτε ότι έχετε έκδοση 2.9i και άνω , και δώστε την παράμετρο '-b 2048'.)

3.2 Μέγεθος δίσκου

Ένα δίσκος με C κυλίνδρους, H κεφαλές και S τομείς ανά ίχνος έχει C*H*S τομείς συνολικά και χωρητικότητα C*H*S*512 bytes. Για παράδειγμα, αν η ετικέτα λέει C/H/S=4092/16/63 τότε ο δίσκος έχει 4092*16*63=4124736 τομείς και χωράει 4124736*512=2111864832 bytes (2.11 GB). Κατά σύμβαση, δίνεται C/H/S=16383/16/63 για δίσκους μεγαλύτερους των 8.4 GB, και το μέγεθος το δίσκου δε μπορεί πια να διαβαστεί από τις τιμές C/H/S που αναφέρονται από το δίσκο.

4 Πρόσβαση δίσκου

Για να διαβάσετε ή να γράψετε κάτι από το δίσκο, πρέπει να δείξουμε τη θέση στο δίσκο, δίνοντας για παράδειγμα τον τομέα ή την ενότητα. Αν ο δίσκος είναι SCSI, τότε ο αριθμός του τομέα πηγαίνει κατευθείαν στην εντολή SCSI και ο δίσκος την καταλαβαίνει. Αν ο δίσκος είναι IDE χρησιμοποιώντας LBA, ισχύει το ίδιο. Αλλά αν ο δίσκος είναι παλιός RLL ή MFM ή IDE της προ-LBA εποχής, τότε ο δίσκος περιμένει ένα τριπλό αριθμό (κύλινδρο, κεφαλή, τομέα) για να καταδείξει το σημείο.

Η αντιστοιχία μεταξύ γραμμικής διεύθυνσης και της 3D σημειογραφίας είναι: για ένα δίσκο με C κυλίνδρους, H κεφαλές και S τομείς/ίχνος ή θέση (c,h,s) σε 3D ή CHS είναι η ίδια θέση με c*H*S + h*S + (s-1) σε γραμμική ή LBA. (Το μείον ένα είναι επειδή οι τομείς αρχίζουν κατά παράδοση από το 1, όχι το 0 όπως στο 3D).

Κατά συνέπεια, για να πρόσβαση σε έναν πολύ παλιό ο μη-SCSI δίσκο, πρέπει να ξέρουμε την *γεωμετρία*, δηλαδή, τις τιμές C, H και S.

4.1 Πρόσβαση του BIOS και το όριο των 1024 κυλίνδρων

Το Linux δε χρησιμοποιεί το BIOS, αλλά άλλα συστήματα το χρησιμοποιούν. Το BIOS, που προϋπάρχει του LBA, προσφέρει τις ρουτίνες δίσκου INT13 που δέχονται (c,h,s) παραμέτρους. (Ακριβέστερα: το `AH` διαλέγει τη λειτουργία που θα εκτελεστεί, το `CH` είναι τα κάτω 8 bits του ονόματος του κυλίνδρους, το `CL` έχει στα bits 7-6 τα άνω δυο bits του αριθμού του κυλίνδρου και στα bits 5-0 τον αριθμό του τομέα, `DH` είναι ο αριθμός της κεφαλής, και `DL` είναι ο αριθμός του δίσκου (80h ή 81h). Αυτό εξηγεί μερικά τη διάταξη του πίνακα κατατμήσεων.)

Έτσι, έχουμε το CHS κωδικοποιημένο σε 3 bytes, με 10 bits για το όνομα του κυλίνδρου , 8 bits για την κεφαλή και 6 bits για τον αριθμό ίχνους τομέα (1-63). Εξυπακούεται ότι οι κύλινδροι μπορεί να είναι από 0 έως 1023 και δε μπορούν να αριθμηθούν πάνω από 1024 κύλινδροι από το BIOS.

Το DOS και τα Windows δεν άλλαξαν όταν IDE δίσκοι με υποστήριξη LBA εμφανίστηκαν, οπότε το DOS και τα Windows συνέχισα να χρειάζονται γεωμετρία δίσκου, ακόμα κι όταν αυτό δε χρειαζόταν από το I/O του δίσκου, αλλά μόνο για να επικοινωνούν με το BIOS. Αυτό ξανά σημαίνει ότι το Linux χρειάζεται τη γεωμετρία όπου επικοινωνία με το BIOS ή με άλλα λειτουργικά απαιτείται, ακόμα και σε μοντέρνους δίσκους.

Αυτή η κατάσταση κράτησε για τέσσερα χρόνια περίπου και μετά εμφανίστηκαν στην αγορά δίσκοι που δε μπορούσαν να κληθούν με τις συναρτήσεις INT13 (καθότι τα 10+8+6=24 bits για (c,h,s) δεν μπορούν να αριθμήσουν πάνω από 8.5 GB) και μια νέα διασύνδεση με το BIOS σχεδιάστηκε: οι αποκαλούμενες Extended INT13 συναρτήσεις, όπου το DS:SI δείχνει στο 16-byte Disk Address Packet που περιλαμβάνει έναν 8μπιτο αριθμό ενοτήτων.

Πολύ αργά ο κόσμος της Microsoft κινείται προς τη χρήση αυτών των Extended INT13 συναρτήσεων. Μάλλον σε μερικά χρόνια από σήμερα, κανένα μοντέρνο σύστημα δε θα χρειάζεται τη γεωμετρία του δίσκου.

4.2 Ιστορία του BIOS και των ορίων του IDE

ATA Specification (για IDE δίσκους) - το όριο των 137 GB

Το πολύ 65536 κύλινδροι (αριθμημένοι 0-65535), 16 κεφαλές (αριθμημένες 0-15), 255 τομείς/ίχνος (αριθμημένοι 1-255) για μια μέγιστη χωρητικότητα 267386880 τομέων (512 bytes ο καθένας), δηλαδή, 136902082560 bytes (137 GB). Αυτό ακόμα δεν είναι πρόβλημα (το 1999), αλλά θα είναι σε μερικά χρόνια από σήμερα.

BIOS Int 13 - το όριο των 8.5 GB

Το πολύ 1024 κύλινδροι (0-1023), 256 κεφαλές (0-255), 63 τομείς/ίχνος (1-63) για μέγιστη χωρητικότητα 8455716864 bytes (8.5 GB). Αυτός είναι ένας αρκετά σοβαρός περιορισμός σήμερα. Σημαίνει ότι το DOS δε μπορεί να χρησιμοποιήσει τους νέους μεγάλους δίσκους.

Το όριο των 528 MB

Αν οι ίδιες τιμές c,h,s χρησιμοποιούνται για το BIOS Int 13 call και για το I/O του I DE δίσκου, και οι δυο περιορισμοί συνδυάζονται και μπορούμε να χρησιμοποιήσουμε το πολύ 1024 κυλίνδρους, 16 κεφαλές και 63 τομείς/ίχνος για μέγιστη τελική χωρητικότητα 528482304 bytes (528MB), το περίφημο όριο των 504 MiB για DOS με παλιό BIOS. Αυτό έγινε πρόβλημα το 1993 και εφευρέθηκαν πολλά τεχνάσματα, και σε υλικό (LBA), και firmware (μεταφράζοντας το BIOS) και σε software (διαχειριστές δίσκων). Η έννοια της 'μετάφρασης' εφευρέθηκε (1994): το BIOS μπορούσε να χρησιμοποιεί μία γεωμετρία όταν επικοινωνούσε με το δίσκο και άλλη, ψευδή, γεωμετρία όταν μιλούσε στο DOS, και να μεταφράσει μεταξύ τους.

Το όριο των 2.1 GB (Απρίλιος 1996)

Μερικά παλιά BIOS χρησιμοποιούν μόνο 12 bits για το πεδίο στη CMOS RAM που δίνει τον αριθμό των κυλίνδρων. Κατά συνέπεια, ο αριθμός αυτός μπορεί να είναι το πολύ 4095, και μόνο 4095*16*63*512=2113413120 bytes είναι προσβάσιμα. Το να υπάρχει μεγαλύτερος δίσκος έχει ως αποτέλεσμα το κρέμασμα κατά την εκκίνηση. Αυτό έκανε δίσκους με γεωμετρία 4092/16/63 αρκετά δημοφιλής. Ακόμα και σήμερα υπάρχουν μεγάλοι δίσκοι που έρχονται με διακόπτη για εμφανίζονται ως 4092/16/63. Δείτε και το *over2gb.htm.*

Το όριο των 3.2 GB

Υπήρχε ένα μεγάλο bug στο Phoenix 4.03 και 4.04 BIOS που τα έκανε να κολλάνε στο CMOS setup για δίσκους μεγαλύτερους των 3227MB. Δείτε το *over3gb.htm.*

Το όριο των 4.2 GB (Φεβρουάριος 1997)

Η απλή μετάφραση του BIOS (ECHS=Extended CHS, μερικές φορές λέγεται και 'Large disk support' ή απλά 'Large') λειτουργεί με τον συνεχή διπλασιασμό του αριθμού των κεφαλών και τον υποδιπλασιασμό του αριθμού των κυλίνδρων που δείχνονται στο DOS, μέχρι οι κύλινδροι να είναι το πολύ 1024. Το DOS και τα Windows 95 δε μπορούν να διαχειριστούν 256 κεφαλές, και στην περίπτωση που ο δίσκος αναφέρει 16 κεφαλές, αυτό σημαίνει ότι ο απλός αυτός μηχανισμός μπορεί να δουλέψει για μέχρι 8192*16*63*512=4227858432 bytes (με ψευδή γεωμετρία με 1024 κυλίνδρους, 128 κεφαλές, 63 τομείς/ίχνος). Σημειώστε ότι το ECHS δεν αλλάζει τον αριθμό των τομέων ανά ίχνος, οπότε αν δεν είναι 63, το όριο θα είναι ακόμα χαμηλότερο. Δείτε το *over4gb.htm.*

Το όριο των 7.9 GB

Λίγο πιο έξυπνα BIOS αποφεύγουν αυτό το πρόβλημα με τη ρύθμιση πρώτα του αριθμού των κεφαλών σε 15 ('revised ECHS'), ώστε η ψευδής γεωμετρία να διατηρείται με 240 κεφαλές, αρκετό για 1024*240*63*512=7927234560 bytes.

Το όριο των 8.4 GB

Τελειώνοντας, αν το BIOS κάνει ό,τι μπορεί για μια επιτυχή μετάφραση, χρησιμοποιεί 255 κεφαλές με 63 τομείς/ίχνος ('assisted LBA' or just 'LBA') και φτάνει τα 1024*255*63*512=8422686720 bytes, λίγο μικρότερο από το προηγούμενο όριο των 8.5 GB, εφόσον γεωμετρίες με 256 κεφαλές πρέπει να αποφεύγονται. (Η μετάφραση θα χρησιμοποιήσει για τον αριθμό κεφαλών τον αριθμό H από την ακολουθία 16, 32, 64, 128, 255 για την οποία η συνολική χωρητικότητα φτάνει στα 1024*H*63*512, και μετά υπολογίζει τον αριθμό των κυλίνδρων C ως την χωρητικότητα διαιρούμενη με (H*63*512).)

The 33.8 GB limit (August 1999)

Το επόμενο εμπόδιο έρχεται με μεγέθη άνω των 33.8 GB. Το πρόβλημα είναι ότι με 16 κεφαλές και 63τομείς/ίχνος αυτό αντιστοιχεί σε αριθμό κυλίνδρων πάνω από 65535, που δε χωράει σε short αριθμό. Τα περισσότερα BIOS σήμερα δε μπορούν να χειριστούν τέτοιους δίσκους. (Δείτε *Asus upgrades* για νέες εκδόσεις που δουλεύουν.) Πυρήνες παλαιότεροι των 2.2.14 / 2.3.21 χρειάζονται patch. Δείτε 12.1 (IDE προβλήματα με 34+ GB δίσκους) παρακάτω.

Για περαιτέρω συζήτηση αυτού του θέματος δείτε *Breaking the Barriers* και, για περισσότερες λεπτομέρειες, *IDE Hard Drive Capacity Barriers*.

Δίσκοι μεγαλύτεροι των 8.4 GB αναφέρουν τη γεωμετρία τους ως 16383/16/63. Αυτό σημαίνει ότι η 'γεωμετρία' είναι ανεπαρκής και ότι η συνολική χωρητικότητα δε μπορεί να υπολογιστεί από τη γεωμετρία.

5 Εκκίνηση

Όταν το σύστημα ξεκινά, το BIOS διαβάζει τον τομέα 0 (γνωστός και ως MBR - Master Boot Record) από τον πρώτο δίσκο (ή από δισκέτα ή CD-ROM) και διαβάζει τον κώδικα που βρίσκει εκεί - συνήθως έναν φορτωτή. Αυτά τα προγραμματάκια συνήθως δεν περιέχουν οδηγούς και χρησιμοποιούν το BIOS. Αυτό σημαίνει ότι ο πυρήνας του Linux μπορεί να φορτωθεί μόνο όταν βρίσκεται ολόκληρος στους πρώτους 1024 κυλίνδρους.

Το πρόβλημα αυτό λύνεται πολύ εύκολα: σιγουρευτείτε ότι ο πυρήνας (και ίσως και άλλα αρχεία που χρειάζονται κατά την εκκίνηση, όπως τα αρχεία του LILO) βρίσκονται σε μια κατάτμηση που περιέχεται εξ ολοκλήρου στους πρώτους 1024 κυλίνδρους και ότι το BIOS μπορεί να έχει πρόσβαση - αυτό σημαίνει τον πρώτο ή δεύτερο δίσκο.

Έτσι: δημιουργήστε μια κατάτμηση, ας πούμε 10MB, ώστε να υπάρχει χώρος για μερικούς πυρήνες, σιγουρεύοντας ότι βρίσκεται ολόκληρη στους πρώτους 1024 κυλίνδρους του πρώτου ή δεύτερου δίσκου. Προσαρτήστε την στο `/boot` ώστε το LILO να βάλει ό,τι χρειάζεται εκεί.

5.1 Το LILO και η επιλογή 'linear'

Άλλο ένα σημείο στο οποίο ο φορτωτής και το BIOS πρέπει να συμφωνούν είναι η γεωμετρία του δίσκου. Το LILO ρωτά τον πυρήνα για τη γεωμετρία, αλλά όλο και περισσότεροι προγραμματιστές οδηγών έχουν την κακή συνήθεια να παίρνουν τη γεωμετρία από τον πίνακα κατατμήσεων, αντί να λένε στο LILO τι θα χρησιμοποιεί το BIOS. Έτσι, η γεωμετρία από τον πυρήνα είναι συχνά άχρηστη. Σε αυτές τις περιπτώσεις είναι χρήσιμο να βάλετε στο LILO την επιλογή '`linear`'. Το αποτέλεσμα είναι ότι το LILO δε χρειάζεται τη γεωμετρία κατά την εγκατάσταση του φορτωτή αλλά κάνει τη μετατροπή της γραμμικής διεύθυνσης κατά την εκκίνηση. Και γιατί αυτό δεν ισχύει εξ ορισμού; Υπάρχει ένα μειονέκτημα: με την επιλογή 'linear' το LILO δεν ξέρει για τον αριθμό των κυλίνδρων, οπότε δεν μπορεί να σας προειδοποιήσει αν μέρος του πυρήνα είναι εγκατεστημένο μετά το πέρας των 1024 κυλίνδρων και μπορεί να καταλήξετε με ένα σύστημα που δεν ξεκινά.

5.2 Ένα bug του LILO

Με εκδόσεις του LILO κάτω του v21 υπάρχει ένα ακόμα πρόβλημα: η μετατροπή διευθύνσεων που γίνεται κατά την εκκίνηση είναι προβληματική: όταν το c*H είναι 65536 ή παραπάνω, δημιουργείται λάθος στον υπολογισμό. Για H μεγαλύτερα του 64 δημιουργείται αυστηρότερο όριο για το c από το γνωστό c < 1024; για παράδειγμα, με H=255 και ένα παλιό LILO πρέπει να έχετε c < 258. (c=ο κύλινδρος όπου βρίσκεται ο πυρήνας, H=αριθμός κεφαλών).

5.3 Οι 1024 κύλινδροι δεν είναι 1024 κύλινδροι

Ο Tim Williams γράφει: 'Είχα την κατάτμηση του Linux στους πρώτους 1024 κυλίνδρους και πάλι δεν ξεκινούσε. Μόνο όταν το έβαλα πριν το 1 GB δούλεψε'. Πώς γίνεται αυτό; Αυτός ήταν ένας SCSI δίσκος με AHA2940UW ελεγκτή που χρησιμοποιεί είτε H=64, S=32 (δηλαδή κύλινδροι του 1 MiB = 1.05 MB), ή H=255, S=63 (δηλαδή κύλινδροι των 8.2 MB), ανάλογα με τις επιλογές στον δίσκο και το BIOS. Αναμφισβήτητα το BIOS υποθέτει το πρώτο, οπότε οι 1024 κύλινδροι φτάνουν μέχρι το 1 GiB, ενώ το Linux χρησιμοποιεί το δεύτερο και το LILO νόμιζε ότι το όριο ήταν στα 8.4 GB.

6 Γεωμετρία δίσκου, κατατμήσεις και 'επικαλύψεις'

Εάν έχετε αρκετά λειτουργικά συστήματα στους δίσκους σας, τότε καθένα χρησιμοποιεί μία ή περισσότερες κατατμήσεις. Μια ασυμφωνία για το που βρίσκονται αυτές οι κατατμήσεις θα έχει καταστροφικά αποτελέσματα.

Το MBR περιέχει έναν *πίνακα κατατμήσεων* που περιγράφει που βρίσκονται οι (πρωταρχικές) κατατμήσεις. Υπάρχουνε 4 εγγραφές για 4 πρωταρχικές κατατμήσεις, με κάθε μία να είναι

```
struct partition {
        char active;    /* 0x80: bootable, 0: not bootable */
        char begin[3];  /* CHS for first sector */
        char type;
        char end[3];    /* CHS for last sector */
        int start;      /* 32 bit sector number (counting from 0) */
        int length;     /* 32 bit number of sectors */
};
```

(όπου CHS είναι Cylinder/Head/Sector).

Αυτές οι πληροφορίες είναι περιττές: η περιοχή της κατάτμησης δίνεται και από το πεδία των 24-bit `begin` και `end`, και από τα πεδία των 32-bit `start` και `length`.

Το Linux χρησιμοποιεί μόνο τα πεδία `start` και `length` και, έτσι, μπορεί να διαχειριστεί κατατμήσεις με το πολύ 2^32 τομείς, δηλαδή, κατατμήσεις το πολύ 2 TiB. Αυτό είναι 100 φορές περισσότερο από τους σημερινούς δίσκους, οπότε μάλλον θα είναι αρκετό για τα επόμενο 8 περίπου χρόνια. (Έτσι, οι κατατμήσεις μπορεί να είναι πολύ μεγάλες, αλλά υπάρχει ο σοβαρός περιορισμός ότι σε ένα ext2 σύστημα αρχείων σε μηχάνημα με 32-bit ακεραίους ένα αρχείο δε μπορεί να είναι μεγαλύτερο από 2 GiB.)

Το DOS χρησιμοποιεί τα `begin` και `end` πεδία, και χρησιμοποιεί την BIOS INT13 κλήση για πρόσβαση στο δίσκο, και έτσι μπορεί να δει δίσκους το πολύ 8.4GB, ακόμα και με BIOS που κάνει μετάφραση. (Οι κατατμήσεις δε μπορούν να είναι πάνω από 2.1 GB λόγω περιορισμών του FAT16 συστήματος). Το ίδιο ισχύει και για τα Windows 3.11 και WfWG και Windows NT 3.* και Novell NetWare.

Τα Windows 95 έχουν υποστήριξη για το Extended INT13, και χρησιμοποιούν ειδικούς τύπους κατατμήσεων (c, e, f αντί για b, 6, 5) για να δείξουν ότι η κατάτμηση θα χρησιμοποιηθεί έτσι. Όταν αυτοί οι τύποι κατατμήσεων χρησιμοποιούνται, τα πεδία `begin` και `end` περιέχουν ψεύτικα στοιχεία (1023/255/63). Τα Windows 95 OSR2 εισήγαγαν το FAT32 σύστημα (τύποι κατατμήσεων b or c), που επιτρέπει κατατμήσεις το πολύ 2 TiB.

Τι είναι αυτά που σας δείχνει το `fdisk` για 'επικαλυπτόμενες' κατατμήσεις, όταν στην πραγματικότητα όλα είναι εντάξει; Υπάρχει κάτι 'λάθος': αν δείτε τα `begin` και `end` πεδία τέτοιων κατατμήσεων, όπως κάνει το DOS, επικαλύπτονται. (Και αυτό δε μπορεί να διορθωθεί, αφού τα πεδία αυτά δε μπορούν να αποθηκεύσουν αριθμούς κυλίνδρων άνω του 1024 - θα υπάρχει πάντα 'επικάλυψη' όταν έχετε περισσότερους από 1024 κυλίνδρους.). Παρόλα αυτά, αν δείτε τα `start` και `length` πεδία, όπως κάνει το Linux, και τα Windows 95 στην περίπτωση κατατμήσεων με τύπο c, e ή f, τότε όλα είναι εντάξει. Έτσι, αγνοήστε τις προειδοποιήσεις του όταν το `cfdisk` είναι ικανοποιημένο και έχετε ένα δίσκο μόνο με Linux. Προσέξτε όταν ο δίσκος μοιράζεται με το DOS. Χρησιμοποιήστε τις εντολές `cfdisk -Ps /dev/hdx` και `cfdisk -Pt /dev/hdx` για να δείτε τον πίνακα κατατμήσεων του `/dev/hdx`.

7 Μετάφραση και Διαχειριστές Δίσκων

Η γεωμετρία του δίσκου (με κεφαλές, κυλίνδρους και ίχνη) είναι κάτι από την εποχή του MFM και του RLL. Εκείνες τις μέρες αυτή ήταν η πραγματικότητα. Σήμερα, με τα IDE ή SCSI, κανείς δεν ενδιαφέρεται ποια είναι η 'πραγματική' γεωμετρία τους δίσκου. Ακόμη, ο αριθμός των τομέων ανά ίχνος είναι μεταβλητός: υπάρχουν περισσότεροι τομείς στο εξωτερικό του δίσκου και έτσι δεν υπάρχει 'πραγματικός' αριθμός τομέων ανά ίχνος. Αντιθέτως: η IDE εντολή INITIALIZE DRIVE PARAMETERS (91h) χρησιμοποιείται για να λέει στον δίσκο πόσες κεφαλές και τομείς/ίχνος υποτίθεται ότι έχει. Είναι αρκετά συχνό να δείτε ένα μεγάλο μοντέρνο δίσκο με 2 κεφαλές να αναφέρει 15 ή 16 κεφαλές στο BIOS, ενώ το BIOS να αναφέρει 255 κεφαλές στα προγράμματα.

Για τον χρήστη είναι καλύτερο να έχει το δίσκο ως ένα γραμμικό σύνολο τομέων αριθμημένους ως 0, 1, ..., και να αφήσει τα ηλεκτρονικά να βρουν που βρίσκεται ο κάθε τομέας στον δίσκο. Αυτή η γραμμική αρίθμηση λέγεται LBA.

Έτσι η γενική εικόνα είναι ως εξής: Το DOS, ή κάποιος άλλος φορτωτής, μιλά στο BIOS, αναφέροντας τα (c,h,s). Το BIOS τα μετατρέπει σε LBA χρησιμοποιώντας την ψεύτικη γεωμετρία που χρησιμοποιεί ο χρήστης. Αν ο δίσκος δεχτεί το LBA τότε αυτή η τιμή χρησιμοποιείται για την πρόσβαση. Αλλιώς, μετατρέπεται πίσω σε (c',h',s') χρησιμοποιώντας τη γεωμετρία που αναφέρει ο δίσκος και έτσι γίνεται η πρόσβαση.

Σημειώστε ότι υπάρχει κάποια σύγχυση στην χρήση του 'LBA': Σαν όρος που περιγράφει τις δυνατότητες του δίσκου σημαίνει 'Γραμμική Διευθυνσιοδότηση Τεμαχίων = Linear Block Addressing' (σε αντίθεση με τη διευθυνσιοδότηση CHS). Σαν όρος στο BIOS Setup, περιγράφει ένα είδος μετάφρασης που μερικές φορές καλείται 'βοηθούμενο LBA = assisted LBA' - δείτε παραπάνω το '4.2 ()'.

Κάτι παρόμοιο συμβαίνει όταν ο δίσκος δεν καταλαβαίνει το LBA αλλά το BIOS ξέρει την μετάφραση. (Στο setup αυτό συνήθως ονομάζεται 'Large'.) Έτσι το BIOS παρουσιάζει γεωμετρία (C,H,S) στο λειτουργικό σύστημα και χρησιμοποιεί (C',H',S') όταν επικοινωνεί με τον ελεγκτή. Συνήθως S = S', C = C'/N και H = H'*N, όπου e N είναι η μεγαλύτερη δύναμη του δύο που σιγουρεύει ότι C' <= 1024 (έτσι η χωρητικότητα στρογγυλεύεται προς τα κάτω σε C' = C/N). Αυτό επιτρέπει πρόσβαση μέχρι 8.4 GB (7.8 GiB).

(Η τρίτη επιλογή στο setup είναι συνήθως η 'Normal', όπου δε γίνεται καμία μετάφραση.)

Εάν το BIOS δεν ξέρει τα 'Large' ή 'LBA', τότε υπάρχουν λύσεις με προγράμματα. Διαχειριστές δίσκων όπως οι OnTrack ή EZ-Drive αντικαθιστούν τις ρουτίνες διαχείρισης του BIOS με τις δικές τους. Αυτό συνήθως επιτυγχάνεται με το να υπάρχει ο κώδικας του διαχειριστή στο MBR και επακόλουθους τομείς (το OnTrack ονομάζει αυτόν τον κώδικα DDO: Dynamic Drive Overlay), ώστε να εκκινείται πριν το λειτουργικό σύστημα. Αυτός είναι ο λόγος που μπορεί να υπάρχουν προβλήματα αν κάποιος ξεκινήσει το μηχάνημα με δισκέτα και υπάρχει και διαχειριστής δίσκου.

Το αποτέλεσμα είναι λίγο-πολύ το ίδιο με τη μετάφραση του BIOS - αλλά αν υπάρχουν διαφορετικά λειτουργικά συστήματα στον ίδιο δίσκο τότε δημιουργούνται πολλά προβλήματα.

Το Linux υποστηρίζει το OnTrack από την έκδοση 1.3.14, και το EZ-Drive από την έκδοση 1.3.29. Περισσότερες λεπτομέρειες δίνονται παρακάτω.

8 Μετάφραση του πυρήνα για δίσκους IDE

Εάν ο πυρήνας ανιχνεύσει την ύπαρξη κάποιου διαχειριστή δίσκου σε έναν IDE δίσκο, θα προσπαθήσει να διαιρέσει τον δίσκο όπως ο διαχειριστής, ώστε το Linux να βλέπει τις ίδιες κατατμήσεις που θα έβλεπε, για παράδειγμα, το DOS με το OnTrack ή το EZ-Drive. Παρόλα αυτά, ΔΕΝ γίνεται διαίρεση όταν η γεωμετρία έχει δηλωθεί στη γραμμή εντολών - έτσι η εντολή '`hd=`*cyls*`,`*heads*`,`*secs*' μπορεί να εξαφανίσει την συμβατότητα με το διαχειριστή δίσκου.

Η αναδιαίρεση γίνεται χρησιμοποιώντας 4, 8, 16, 32, 64, 128, 255 κεφαλές (κρατώντας το H*C σταθερό) μέχρι είτε C <= 1024 ή H = 255.

Οι λεπτομέρειες ακολουθούν - οι υποεπικεφαλίδες είναι τα μυνήματα που εμφανίζονται κατά την εκκίνηση. Εδώ και οπουδήποτε αλλού σε αυτό το κείμενο οι τύποι των κατατμήσεων δίνονται σε δεκαεξαδικά νούμερα.

8.1 EZD

Το EZ-Drive ανιχνεύεται λόγω του ότι η πρώτη πρωταρχική κατάτμηση έχει τύπο 55. Η γεωμετρία διαβάζεται όπως περιγράφεται παραπάνω αντί του πίνακα κατατμήσεων του τομέα 0 - ο πίνακας διαβάζεται από τον τομέα 1. Οι αριθμοί τεμαχίων του δίσκου δεν αλλάζονται, αλλά εγγραφές στον τομέα 0 αναδρομολογούνται στον τομέα 1. Αυτή η συμπεριφορά μπορεί να αλλαχθεί αναμεταλωττίζοντας τον πυρήνα με `#define FAKE_FDISK_FOR_EZDRIVE 0` στο `ide.c`.

8.2 DM6:DDO

Ο OnTrack DiskManager (στον πρώτο δίσκο) ανιχνεύεται από το γεγονός ότι η πρώτη πρωταρχική κατάτμηση έχει τύπο 54. Η γεωμετρία διαβάζεται όπως αναφέρθηκε παραπάνω και ολόκληρος ο δίσκος ((μετακινείται» κατά 63 τομείς (ώστε ο παλιός τομέας 63 να γίνει ο τομέας 0). Μετά, ένα καινούριο MBR (με τον πίνακα κατατμήσεων) διαβάζεται από τον νέο τομέα 0. Φυσικά αυτό γίνεται για να δημιουργηθεί χώρος για το DDO - για αυτό δεν γίνεται αυτή η αλλαγή στους υπόλοιπους δίσκους.

8.3 DM6:AUX

Ο OnTrack DiskManager (στους άλλους δίσκους) ανιχνεύεται από την πρώτη πρωταρχική κατάτμηση που έχει τύπο 51 ή 53. Η γεωμετρία διαβάζεται όπως περιγράφεται παραπάνω.

8.4 DM6:MBR

Μια παλαιότερη έκδοση του OnTrack DiskManager δεν ανιχνεύεται από τον τύπο κατάτμησης αλλά από το αποτύπωμα του. (Ελέγχεται αν η μετατόπιση που βρίσκεται στα πρώτα 2 και 3 bytes του MBR δεν είναι παραπάνω από 430, αν η έλλειψη είναι ίση με 0χ 55AA και αν ακολουθείται από μονό byte). Ξανά η γεωμετρία διαβάζεται όπως παραπάνω.

8.5 PTBL

Τέλος, υπάρχει ένας έλεγχος που προσπαθεί να βρει τη μετάφραση από τις τιμές `start` και `end` των πρωταρχικών κατατμήσεων: Εάν κάποια κατάτμηση έχει αρχικό και τελικό τομέα 1 και 63 αντίστοιχα και έχει τελικές κεφαλές 31, 63, 127 ή 254, τότε, εφόσον συνήθως οι κατατμήσεις τελειώνουν στα όρια του κυλίνδρου, και, επίσης, το IDE υποστηρίζει το πολύ 16 κεφαλές, συμπεραίνεται ότι το BIOS μεταφράζει και η γεωμετρία αλλάζει για να χρησιμοποιηθούν 32, 64, 128 ή 255 κεφαλές αντίστοιχα. Παρόλα αυτά, δεν γίνεται καμία αλλαγή όταν η παρόν γεωμετρία έχει ήδη 63 τομείς ανά ίχνος και τουλάχιστον 63 κεφαλές (το οποίο ότι έχει ήδη γίνει μια αλλαγή γεωμετρίας).

9 Συνέπειες

Τι σημαίνουν όλα αυτά; Για τους χρήστες του Linux μόνο ένα πράγμα: πρέπει να σιγουρευτούν ότι το LILO και το `fdisk` χρησιμοποιούν τη σωστή γεωμετρία, όπου ((σωστή» για το `fdisk` είναι η γεωμετρία που χρησιμοποιείται και από τα άλλα λειτουργικά στον ίδιο δίσκο, και για το LILO αυτή που θα επιτρέψει τη σωστή επικοινωνία με το BIOS κατά την εκκίνηση (συνήθως αυτά τα δυο συμπίπτουν).

Πώς ξέρει το `fdisk` για τη γεωμετρία; Ρωτά τον πυρήνα, χρησιμοποιώντας το `HDIO_GETGEO` ioctl, πριν ο χρήστης επέμβει στη γεωμετρία.

Πώς ξέρει το LILO τη γεωμετρία; Ρωτά την πυρήνα χρησιμοποιώντας το `HDIO_GETGEO` ioctl. Αλλά ο χρήστης μπορεί να επέμβει χρησιμοποιώντας την επιλογή '`disk=`' στο `/etc/lilo.conf` (δείτε το lilo.conf(5)). Μπορείτε να δώσετε και την `linear` επιλογή στο LILO, και θα αποθηκεύσει LBA διευθύνσεις αντί για CHS στον χάρτη του, και θα βρει τη γεωμετρία κατά την εκκίνηση (χρησιμοποιώντας την INT 13 Function 8 για να ρωτήσει για τη γεωμετρία).

Πώς ξέρει ο πυρήνας τί να απαντήσει; Πρώτα απ' όλα, χρήστης μπορεί να έχει δηλώσει τη γεωμετρία με την εντολή '`hda=`*cyls*`,`*heads*`,`*secs*' στον πυρήνα (δείτε bootparam(7)), ίσως χειροκίνητα ή ζητώντας τον boot loader να δώσει αυτή την παράμετρο στον πυρήνα. Για παράδειγμα μπορείτε να πείτε στο LILO να δώσει μια τέτοια παράμετρο προσθέτοντας το '`append = "hda=`*cyls*`,`*heads*`,`*secs*`"`' στο `/etc/lilo.conf` (δείτε lilo.conf(5)). Διαφορετικά ο πυρήνας θα μαντέψει, πιθανόν χρησιμοποιώντας τιμές που βρήκε από το BIOS ή τον δίσκο.

Είναι δυνατόν (από τον πυρήνα 2.1.79) να αλλάξετε τη γεωμετρία στον πυρήνα χρησιμοποιώντας το `/proc`. Για παράδειγμα

```
# sfdisk -g /dev/hdc
```

```
/dev/hdc: 4441 cylinders, 255 heads, 63 sectors/track
# cd /proc/ide/ide1/hdc
# echo bios_cyl:17418 bios_head:128 bios_sect:32 > settings
# sfdisk -g /dev/hdc
/dev/hdc: 17418 cylinders, 128 heads, 32 sectors/track
#
```

9.1 Υπολογίζοντας τις παραμέτρους του LILO

Μερικές φορές είναι χρήσιμο να ορίσετε τη γεωμετρία χρησιμοποιώντας το '`hda=`*cyls*`,`*heads*`,`*secs*' στη γραμμή εντολών του πυρήνα. Συνήθως πάντα χρειάζεται *secs*=63, και ο λόγος που το προθέτουμς είναι για να οριστούν οι *heads*. (Λογικές τιμές σήμερα είναι *heads*=16 και *heads*=255.) Τι θα πρέπει να ορίσουμε για το *cyls*? Ακριβώς τον αριθμό που θα δώσει τη σωστή συνολική χωρητικότητα για C*H*S τομείς. Για παράδειγμα, για ένα δίσκο με 71346240 τομείς (36529274880 bytes) το C υπολογίζεται ως 71346240/(255*63)=4441 (για παράδειγμα χρησιμοποιώντας ένα πρόγραμμα σαν το `bc`), και η παράμετρος εκκίνησης είναι `hdc=4441,255,63`. Πώς ξέρουμε την σωστή χωρητικότητα; Για παράδειγμα,

```
# hdparm -g /dev/hdc | grep sectors
 geometry     = 4441/255/63, sectors = 71346240, start = 0
# hdparm -i /dev/hdc | grep LBAsects
 CurCHS=16383/16/63, CurSects=16514064, LBA=yes, LBAsects=71346240
```

δίνει δύο τρόπους να βρούμε τον συνολικό αριθμό τομέων 71346240. Ο πυρήνας μας δίνει

```
# dmesg | grep hdc
...
hdc: Maxtor 93652U8, 34837MB w/2048kB Cache, CHS=70780/16/63
 hdc: [PTBL] [4441/255/63] hdc1 hdc2 hdc3! hdc4 < hdc5 > ...
```

που μας λέει για (τουλάχιστον) 34837*2048=71346176 και για (τουλάχιστον) 70780*16*63=71346240 τομείς. Σε αυτή την περίπτωση, η δεύτερη τιμή συμβαίνει να είναι σωστή, αλλά γενικά και οι δύο μπορούν να στρογγυλοποιηθουν προς τα κάτω. Αυτός είναι ένας καλός τρόπος να προσεγγίσουμε το μέγεθος του δίσκου όταν το `hdparm` δεν είναι διαθέσιμο. Ποτέ μην δίνεται πολύ μεγάλη τιμή για το *cyls*! Στην περίπτωση των SCSI δίσκων ο ακριβής αριθμός των τομέων δίνεται κατά την εκκίνηση:

```
SCSI device sda: hdwr sector= 512 bytes. Sectors= 17755792 [8669 MB] [8.7 GB]
```

(και τα MB, GB είναι στρογγυλοποιημένα, όχι προς τα κάτω, και 'δυαδικά').

10 Λεπτομέρειες

10.1 IDE - οι επτά γεωμετρίες

Ο IDE οδηγός έχει πέντε πηγές πληροφοριών για τη γεωμετρία. Η πρώτη (G_user) είναι αυτή που δηλώνεται από τον χρήστη στη γραμμή εντολών. Η δεύτερη (G_bios) είναι ο πίνακας παραμέτρων δίσκων του BIOS (Fixed Disk Parameter Table) (για τον πρώτο και δεύτερο δίσκο μόνο) που διαβάζεται κατά την εκκίνηση του συστήματος, πριν την αλλαγή σε λειτουργία 32-bit. Η τρίτη (G_phys) και τέταρτη (G_log) επιστρέφονται από τον ελεγκτή IDE ως απάντηση στην εντολή IDENTIFY - είναι η ‘φυσική’ και ‘τρέχουσα λογική’ γεωμετρία αντίστοιχα.

Από την άλλη, ο οδηγός χρειάζεται δύο τιμές για τη γεωμετρία: από τη μία την G_fdisk, που επιστρέφεται από το `HDIO_GETGEO` ioctl, και από την άλλη την G_used, που χρησιμοποιείται για το πραγματικό I/O. Και οι δύο χρησιμοποιούν το G_user εάν έχει δοθεί, το G_bios όταν αυτές οι πληροφορίες είναι διαθέσιμες σύμφωνα με το CMOS, ή το G_phys αν κανένα από τα προηγούμενα δεν είναι διαθέσιμο. Εάν το G_log φαίνεται λογικό τότε το G_used ρυθμίζεται σε αυτό. Διαφορετικά, αν το G_used δεν είναι σωστό και το G_phys φαίνεται λογικό χρησιμοποιείται το G_phys για να ρυθμιστεί το G_used. ‘Λογικό’ σημαίνει ότι ο αριθμός των κεφαλών είναι 1-16.

Με άλλα λόγια: η γραμμή εντολών υπερβαίνει το BIOS, και θα καθορίσει τί θα δει το `fdisk`, αλλά αν καθορίζει την μεταφραζόμενη γεωμετρία (με παραπάνω από 16 κεφαλές), για το I/O του πυρήνα θα χρησιμοποιηθούν οι τιμές της εντολής IDENTIFY.

Σημειώστε ότι το G_bios είναι αρκετά αναξιόπιστο: για συστήματα που ξεκινάνε με SCSI ο πρώτος και δεύτερος δίσκος μπορούν να είναι SCSI και η γεωμετρία που αναφέρει το BIOS για sda μπορεί να χρησιμοποιηθεί από τον πυρήνα για τα hda. Επίσης, οι δίσκοι που δεν αναφέρονται στο setup του BIOS δεν βλέπονται από το BIOS. Αυτό σημαίνει ότι, π.χ., σε ένα σύστημα με IDE δίσκους μόνο, αν το hdb δε δοθεί στο BIOS setup, η γεωμετρία που αναφέρεται από το BIOS για τον πρώτο και δεύτερο δίσκο θα χρησιμοποιηθεί για τα hda και hdc.

10.2 SCSI λεπτομέρειες

Στην περίπτωση του SCSI τα πράγματα είναι λίγο διαφορετικά, αφού οι SCSI εντολές ήδη χρησιμοποιούν λογικούς αριθμούς, ώστε η ‘γεωμετρία’ είναι τελείως άσχετη από το I/O. Παρόλα αυτά, ο πίνακας κατατμήσεων είναι ίδιος, και έτσι το `fdisk` πρέπει να βρει τη γεωμετρία, ενώ χρησιμοποιεί και το `HDIO_GETGEO`. Ακόμη, το `fdisk` δεν διαχωρίζει μεταξύ των IDE και SCSI δίσκων. Όπως θα δείτε από την παρακάτω περιγραφή, οι διάφοροι οδηγοί χρησιμοποιούν ο καθένας κάπως διαφορετική γεωμετρία. Αρκετά μπερδεμένη κατάσταση.

Αν δεν χρησιμοποιείτε το DOS, τότε αποφύγετε όλες τις ρύθμισης των μεταφράσεων, χρησιμοποιείστε 64 κεφαλές, 32 τομείς/ίχνος (για ένα βολικό 1 MiB ανά κύλινδρο), αν είναι δυνατόν, ώστε να μην έχετε προβλήματα όταν μεταφέρετε τον δίσκο από τον έναν ελεγκτή στον άλλο. Μερικοί SCSI οδηγοί (aha152x, pas16, ppa, qlogicfas, qlogicisp) για να διατηρήσουν συμβατότητα με το DOS δε θα σας επιτρέψουν να χρησιμοποιήσετε πάνω από 8 GiB ακόμα και σε σύστημα με μόνο Linux. Αυτό είναι bug.

Ποια είναι η πραγματική γεωμετρία; Η ευκολότερη απάντηση είναι ότι δεν υπάρχει. Ακόμα κι αν υπήρχε, δε θέλετε να την ξέρετε και σίγουρα ΠΟΤΕ δε θα πείτε στο `fdisk` ή το LILO ή τον πυρήνα ποια είναι. Είναι καθαρά μεταξύ του SCSI ελεγκτή και του δίσκου. Να το επαναλάβω: μόνο κάποιος ανόητος λέει στα `fdisk`/LILO/kernel την πραγματική γεωμετρία ενός SCSI δίσκου.

Αλλά αν είστε περίεργοι και επιμένετε μπορείτε να ρωτήσετε τον δίσκο. Υπάρχει η σημαντική εντολή READ CAPACITY που θα δώσει το μέγεθος του δίσκου και υπάρχει και η MODE SENSE εντολή που στη σελίδα Rigid Disk Drive Geometry (04) δίνει τον αριθμό των κεφαλών και των κυλίνδρων (αυτά δε μπορούν να αλλάξουν) και στη σελίδα Format (03) δίνει τον αριθμό των bytes ανά τομέα και τομέων ανά ίχνος. Το τελευταίο νούμερο δεν είναι σταθερό, αφού ο αριθμός τομέων/ίχνος εξαρτάται από την περιοχή του δίσκου: στο εξωτερικό του δίσκου υπάρχουν περισσότεροι τομείς/ίχνος. Το πρόγραμμα `scsiinfo` στο Linux θα σας δώσει αυτές τις πληροφορίες. Υπάρχουν πολλές λεπτομέρειες και επιπλοκές και είναι ξεκάθαρο ότι κανείς (ούτε και το ίδιο το λειτουργικό) δε θέλει να ξέρει. Ακόμη, εφόσον μας ενδιαφέρει το `fdisk` και το LILO, η συνήθεις απαντήσεις είναι του τύπου C/H/S=4476/27/171 - τιμές που δε μπορούν να χρησιμοποιηθούν από το `fdisk` αφού ο πίνακας κατατμήσεων χρησιμοποιεί 10/8/6 bits για τα C/H/S.

Και τότε από που το HDIO_GETGEO βρίσκει τις πληροφορίες; Από τον ελεγκτή SCSI ή κάνοντας μια εκτίμηση. Μερικοί οδηγοί νομίζουν ότι θέλουμε την πραγματική γεωμετρία αλλά, φυσικά, θέλουμε μόνο ό,τι το DOS ή το fdisk του OS/2 (ή το AFDISK της Adaptec, κλπ) χρησιμοποιούν.

Σημειώστε ότι το fdisk του linux χρειάζεται τον αριθμό κεφαλών και τομέων/ίχνος H και S για να μετατρέψει LBA νούμερα σε c/h/s αλλά ο αριθμός των κυλίνδρων C δεν χρειάζεται. Μερικοί οδηγοί χρησιμοποιούν (C,H,S) = (1023,255,63) για να δείξουν ότι η χωρητικότητα του δίσκου είναι τουλάχιστον 1023*255*63 sectors. Αυτό είναι ατυχές, αφού δεν μας δίνει το πραγματικό μέγεθος και θα περιορίσει τα περισσότερα fdisk σε περίπου 8 GiB - ένας αρκετά σοβαρός περιορισμός.

Στην περιγραφή παρακάτω, το M δηλώνει την συνολική χωρητικότητα του δίσκου και C, H, S ο αριθμός των κυλίνδρων, κεφαλών και τομείς/ίχνος. Αρκούν τα H, S αν χρησιμοποιήσουμε το C ως M / (H*S).

Κατά σύμβαση, H=64, S=32.

aha1740, dtc, g_NCR5380, t128, wd7000:

H=64, S=32.

aha152x, pas16, ppa, qlogicfas, qlogicisp:

H=64, S=32 εκτός εάν C > 1024, οπότε H=255, S=63, C = min(1023, M/(H*S)). (Το C συμπτύσσεται και το H*S*C δεν είναι προσέγγιση του M. Αυτό θα μπερδέψει τις περισσότερες εκδόσεις του fdisk.) Το ppa.c χρησιμοποιεί M+1 αντί του M και λέει ότι αυτό είναι λόγω ενός bug στο sd.c, όπου το M είναι εκτός κατά 1.

advansys:

H=64, S=32 εκτός αν C > 1024 και ακόμη η επιλογή '> 1 GB' του BIOS είναι ενεργοποιημένη, οπότε H=255, S=63.

aha1542:

Ρωτήστε τον ελεγκτή ποια από τις δυο μεταφράσεις χρησιμοποιεί και χρησιμοποιήστε είτε H=255, S=63 ή H=64, S=32. Στην πρώτη περίπτωση θα δείτε κατά την εκκίνηση "aha1542.c: Using extended bios translation".

aic7xxx:

H=64, S=32 εκτός αν C > 1024, και είτε η επιλογή "extended" κατά την εκκίνηση δίνεται ή το 'extended' bit χρησιμοποιείται στα SEEPROM ή BIOS, οπότε H=255, S=63. Στο Linux 2.0.36 αυτή η μετάφραση πάντα χρησιμοποιείται αν δε βρεθεί SEEPROM, αλλά στο Linux 2.2.6 αν δε βρεθεί SEEPROM ή μετάφραση χρησιμοποιείται αν ο χρήστης το επιθυμεί, χρησιμοποιώντας την παράμετρο εκκίνησης (αν βρεθεί SEEPROM, η παράμετρος αγνοείται). Αυτό σημαίνει ότι το setup που δουλεύει στο 2.0.36 μπορεί να μην εκκινήσει στο 2.2.6 (και να απαιτεί την επιλογή 'linear' στο LILO, ή την παράμετρο 'aic7xxx=extended' στον πυρήνα).

buslogic:

H=64, S=32 εκτός εάν C >= 1024, και η extended μετάφραση ενεργοποιήθηκε στον ελεγκτή, οπότε αν M < 2^22 τότε H=128, S=32; αλλιώς H=255, S=63. Όμως, αφού γίνει αυτή η επιλογή για (C,H,S), ο πίνακας κατατμήσεων διαβάζεται και αν για τις τρεις πιθανότητες (H,S) = (64,32), (128,32), (255,63) η τιμή τελικό H=H-1 βρεθεί, εκείνο το ζευγάρι (H,S) χρησιμοποιείται, και το μήνυμα "Adopting Geometry from Partition Table" τυπώνεται κατά την εκκίνηση.

fdomain:

Βρείτε τη γεωμετρία στις παραμέτρους δίσκων του BIOS, ή διαβάστε τον πίνακα κατατμήσεων και χρησιμοποιήστε H=τελευταίο H+1, S=τελευταίο S για την πρώτη κατάτμηση, εφόσον είναι άδεια, ή χρησιμοποιήστε H=64, S=32 για M < 2^21 (1 GiB), H=128, S=63 για M < 63*2^17 (3.9 GiB) και H=255, S=63 διαφορετικά.

in2000:

Χρησιμοποιήστε το πρώτο από τα (H,S) = (64,32), (64,63), (128,63), (255,63) που θα δώσει C <= 1024. Στην τελευταία περίπτωση, κόψτε το C σε 1023.

seagate:

Διαβάστε τα C,H,S από τον δίσκο. Εάν το C ή S είναι πολύ μεγάλο, τότε βάλτε S=17, H=2 και διπλασιάστε το H μέχρι C <= 1024. Αυτό σημαίνει ότι το H θα είναι 0 αν M > 128*1024*17 (1.1 GiB). Αυτό είναι bug.

ultrastor and u14_34f:

Ένα από τα τρία (H,S) = (16,63), (64,32), (64,63) χρησιμοποιείται, ανάλογο με τον τρόπο λειτουργίας του ελεγκτή.

Αν ο οδηγός δεν δίνει τη γεωμετρία, μαντεύουμε χρησιμοποιώντας τον πίνακα κατατμήσεων ή χρησιμοποιώντας την συνολική χωρητικότητα.

Κοιτάξτε τον πίνακα κατατμήσεων. Εφόσον, κατά σύμβαση, οι κατατμήσεις τελειώνουν σε όριο κυλίνδρους, μπορούμε, με δεδομένο ότι `όριο = (τελικό C,τελικό H,τελικό S)` για οποιαδήποτε κατάτμηση, απλά βάζουμε H = `τελικό H+1` και S = `τελικό S`. (Θυμηθείτε ότι οι τομείς μετριούνται από το 1.) Με περισσότερες λεπτομέρειες, γίνονται τα παρακάτω. Αν δεν υπάρχει ελεύθερη κατάτμηση, διαλέγουμε την κατάτμηση με το μεγαλύτερο `αρχικό C`. Για αυτή την κατάτμηση, κοιτάμε το `τελικό+1`, υπολογισμένο προσθέτοντας τα `αρχή` και `μήκος` και υποθέτοντας ότι η κατάτμηση τελειώνει σε όριο κυλίνδρου. Αν και οι δυο τιμές συμφωνούν ή αν `τελικό C` = 1023 και `αρχή+μήκος` είναι ακέραιο πολλαπλάσιο του `(τελικό H+1)*τελικό S`, τότε υποθέτουμε ότι η κατάτμηση είναι όντως ευθυγραμμισμένη με το όριο του κυλίνδρου, και βάζουμε H = `τελικό H+1` και S = `τελικό S`. Αν αυτό αποτύχει, είτε επειδή δεν υπάρχουν κατατμήσεις, είτε επειδή έχουν περίεργες τιμές, τότε κοιτάμε πάλι μόνο τη χωρητικότητα του δίσκου M. Αλγόριθμος: βάζουμε H = M/(62*1024) (στρογγυλοποιημένο κατά πάνω), S = M/(1024*H) (στρογγυλοποιημένο πάνω), C = M/(H*S) (στρογγυλοποίηση κάτω). Αυτό έχει το αποτέλεσμα να έχουμε τα (C,H,S) με το C το πολύ 1024 και το S το πολύ 62.

11 Το όριο του Linux I DE των 8 GiB

O Linux IDE οδηγός παίρνει τη γεωμετρία και χωρητικότητα του δίσκου (και άλλα πολλά) χρησιμοποιώντας την κλήση ATA IDENTIFY. Μέχρι πρόσφατα, ο οδηγός δε θα πίστευε την επιστρεφόμενη τιμή της lba χωρητικότητας (lba_capacity), αν ήταν πάνω από 10% από την υπολογιζόμενη με C*H*S. Παρόλα αυτά, οι κατασκευαστές, σε μεγάλους IDE δίσκους (με περισσότερους από 16514064 τομείς) επιστρέφουν τα C=16383, H=16, S=63, για ένα σύνολο 16514064 τομέων (7.8 GB), ανεξαρτήτως του πραγματικού τους μεγέθους, αλλά δίνουν την πραγματική χωρητικότητα ως lba.

Οι πρόσφατοι πυρήνες (2.0.34, 2.1.90) το ξέρουν αυτό και το διορθώνουν. Αν έχετε παλαιότερο πυρήνα και δε θέλετε να αναβαθμιστείτε, και ο πυρήνας βλέπει μόνο 8 GiB σε έναν πολύ μεγαλύτερο δίσκο, δοκιμάστε να αλλάξετε τη ρουτίνα `lba_capacity_is_ok` στο `/usr/src/linux/drivers/block/ide.c` σε κάτι σαν

```
static int lba_capacity_is_ok (struct hd_driveid *id) {
        id->cyls = id->lba_capacity / (id->heads * id->sectors);
        return 1;
}
```

Για μια πιο προσεκτική διόρθωση δείτε το 2.1.90.

11.1 BIOS επιπλοκές

Όπως μόλις ανέφερα, οι μεγάλοι δίσκοι επιστρέφουν C=16383, H=16, S=63 ανεξάρτητα από το πραγματικό τους μέγεθος, ενώ το πραγματικό μέγεθος επιστρέφεται ως LBAcapacity. Μερικά BIOS δεν το αναγνωρίζουν αυτό και μεταφράζουν το 16383/16/63 σε κάτι με λιγότερους κυλίνδρους και περισσότερες κεφαλές, π.χ. 1024/255/63 ή 1027/255/63. Έτσι, ο πυρήνας δε μπορεί να αναγνωρίσει τη γεωμετρία 16383/16/63, αλλά και τις μπερδεμένες εκδόσεις της του BIOS. Από τον πυρήνα 2.2.2 όλα αυτά διορθώθηκαν (χρησιμοποιώντας τα H και S του BIOS και υπολογίζοντας το C = χωρητικότητα/(H*S)). Συνήθως το πρόβλημα λύνεται με το να ρυθμιστεί ο δίσκος ως Normal στο BIOS (ή ακόμη καλύτερο ως None, χωρίς να αναφερθεί καθόλου στο BIOS). Αν αυτό δεν είναι δυνατό επειδή πρέπει να εκκινήσετε από αυτόν ή χρησιμοποιείτε DOS/Windows και η αναβάθμιση σε 2.2.2 ή μεγαλύτερη έκδοση δε γίνεται, χρησιμοποιήστε παραμέτρους εκκίνησης στον πυρήνα.

Εάν το BIOS αναφέρει 16320/16/63, τότε αυτό γίνεται συνή 8ως για να έχουμε 1024/255/63 μετά τη μετάφραση.

Υπάρχει ένα ακόμα πρόβλημα. Αν ο δίσκος είχε χωριστεί σε κατατμήσεις πριν την μετάφραση, τότε ο πυρήνας μπορεί κατά την εκκίνηση να δει τη γεωμετρία που χρησιμοποιείται στον πίνακα κατατμήσεων και να αναφέρει `hda: [PTBL] [1027/255/63]`. Αυτό είναι κακό καθότι ο δίσκος είναι τώρα μόνο 8.4GB. Αυτό διορθώθηκε στον 2.3.21. Ξανά, παράμετροι εκκίνησης στον πυρήνα θα βοηθήσουν.

11.2 Βραχυκυκλωτήρες για επιλογή αριθμού κεφαλών

Πολλοί δίσκοι έχουν βραχυκυκλωτήρες (jumpers) που επιτρέπουν να επιλέξετε μεταξύ γεωμετρία 15 ή 16 κεφαλών. Οι συνήθεις ρυθμίσεις θα σας δώσουν 16 κεφαλές. Μερικές φορές και οι δυο γεωμετρίες δίνουν τον ίδιο αριθμό τομέων, μερικές φορές με 15 κεφαλές δίνονται λιγότεροι τομείς. Υπάρχει ένας καλός λόγος για αυτή την επιλογή: ο Petri Kaukasoina γράφει: 'Ένας 10.1 Gig IBM Deskstar 16 GP (IBM-DTTA-351010) ήταν ρυθμισμένος για 16 κεφαλές αλλά στο παλιό PC (με AMI BIOS) δεν εκκινούσε και έπρεπε να το ρυθμίσω σε 15 κεφαλές. Το hdparm -i λέει ότι RawCHS=16383/15/63 και LBAsects=19807200. Χρησιμοποιώ 20960/15/63 για να έχω τη μέγιστη χωρητικότητα.' Η γεωμετρία 16383/15/63 δεν αναγνωρίζεται από τον πυρήνα, οπότε χρειάζονται παράμετροι εκκίνησης. Για τις ρυθμίσεις δείτε *http://www.storage.ibm.com/techsup/hddtech/hddtech.htm*.

11.3 Βραχυκυκλωτήρες που μειώνουν τη συνολική χωρητικότητα

Πολλοί δίσκοι έχουν βραχυκυκλωτήρες που κάνουν τον δίσκο να φαίνεται μικρότερος. Αρκετά ανόητο, και μάλλον κανένας χρήστης του Linux δε θα θέλει να το χρησιμοποιήσει, αλλά μερικά BIOS κολλάνε με μεγάλους δίσκους. Η συνήθης λύση είναι να κρατήσετε τον δίσκο έξω από το BIOS. Αλλά αυτό γίνεται μόνο αν δεν είναι ο δίσκος εκκίνησης.

Το πρώτο σοβαρό όριο ήταν των 4096 κυλίνδρων (δηλαδή, 16 κεφαλές και 63τομείς/ίχνος, 2.11GB). Για παράδειγμα, ένας Fujitsu MPB3032ATU 3.24 GB δίσκος έχει γεωμετρία 6704/15/63, αλλά μπορεί να εμφανιστεί ως 4092/16/63, και μετά αναφέρει LBA χωρητικότητα 4124736 τομείς, ώστε το λειτουργικό σύστημα να μπορεί να μαντέψει ότι στην πραγματικότητα είναι μεγαλύτερος. Σε αυτή την περίπτωση (με ένα BIOS που κολλάει όταν δει πόσο μεγάλος είναι ο δίσκος στην πραγματικότητα ώστε να χρειάζεται ο περιορισμός) χρειάζονται παράμετροι εκκίνησης για να πείτε στο Linux το μέγεθος του δίσκου.

Οι περισσότεροι δίσκοι μπορούν να εμφανιστούν ως δίσκοι 2GB και μετά να αναφέρουν την κομμένη γεωμετρία ως 4092/16/63 ή 4096/16/63, αλλά ακόμα αναφέρουν την πλήρη LBA χωρητικότητα. Τέτοιοι δίσκοι δουλεύουν σωστά και αναφέρουν την πλήρη χωρητικότητα στο Linux, άσχετα με τις θέσεις των βραχυκυκλωτήρων.

Ένα πιο πρόσφατο όριο είναι 12.1 (το όριο των 33.8 GB). Πυρήνες παλαιότεροι από τον 2.3.21 χρειάζονται patch για να μπορέσουν να χρησιμοποιήσουν μεγαλύτερους IDE δίσκους από 33.8GB. Μερικοί μεγαλύτεροι δίσκοι μπορούν να ρυθμιστούν με βραχυκυκλωτήρες ώστε να εμφανίζονται ως 33.8 GB. Για παράδειγμα, ο IBM Deskstar 37.5 GB (DPTA-353750) μπορεί να ρυθμιστεί ώστε να εμφανίζεται ως 33.8 GB, και μετά αναφέρει γεωμετρία 16383/16/63 όπως όλοι οι μεγάλοι δίσκοι, αλλά LBA χωρητικότητα 66055248 (αντίστοιχα με 65531/16/63 ή 4111/255/63)). Αυτοί, όταν ρυθμιστούν ως 33.8GB, χρειάζονται παραμέτρους για πλήρη χωρητικότητα στο Linux. Δείτε το *the BIOS 33.8 GB limit.*

12 Το όριο των 65535 κυλίνδρων στο linux

Το `HDIO_GETGEO` ioctl επιστρέφει τον αριθμό των κυλίνδρων σε τύπο short. Αυτό σημαίνει ότι αν έχτε πάνω από 65535 κυλίνδρους, ο αριθμός περικόπτεται και (για ένα τυπικό SCSI δίσκο με 1 MiB ανά κύλινδρο) ένας δίσκος 80 GiB θα φαίνεται ως 16 GiB. Εφόσον αναγνωρισθεί αυτό το πρόβλημα, αποφεύγεται εύκολα.

12.1 IDE προβλήματα με δίσκους 34+ GB

Δίσκοι μεγαλύτεροι των 33.8 GB δε δουλεύουν με πυρήνες παλαιότερους του 2.3.21. Οι λεπτομέρειες είναι: Υποθέστε ότι αγοράσατε ένα νέο IBM-DPTA-373420 δίσκο με χωρητικότητα 66835440 τομείς (34.2 GB). Πυρήνες παλαιότεροι του 2.3.21 θα σας πούνε ότι το μέγεθος του δίσκου είναι 769*16*63 = 775152 τομείς (0.4 GB), που είναι λίγο απογοητευτικό. Και δίνοντας τις παραμέτρους hdc=4160,255,63 δε βοηθάει καθόλου - απλά αγνοούνται. Τί συμβαίνει; Η ρουτίνα idedisk_setup() βρίσκει τη γεωμετρία που αναφέρει ο δίσκος (που είναι 16383/16/63) και παρακάμπτει ό,τι δίνει ο χρήστης στη γραμμή εντολών, ώστε τα δεδομένα του χρήστη να χρησιμοποιούνται μόνο για τη γεωμετρία του BIOS. Η ρουτίνα current_capacity() ή idedisk_capacity() υπολογίζει τον αριθμό κυλίνδρων ως 66835440/(16*63)=66305, αλλά αφού αποθηκεύεται σε short αριθμό, γίνεται 769. Εφόσον η lba_capacity_is_ok() κατέστρεψε το id->cyls, κάθε επόμενη κλήση σε αυτό θα είναι λάθος και ο δίσκος θα γίνει 769*16*63. Για πολλούς πυρήνες υπάρχει patch. Για τον 2.0.38 μπορεί να βρεθεί στο *ftp.kernel.org.* Για τον 2.2.12 είναι στο *www.uwsg.indiana.edu.* Οι 2.2.14pre πυρήνες υποστηρίζουν αυτούς τους δίσκους. Στους 2.3.* πυρήνες, υπάρχει υποστήριξη από τον 2.3.21. Πάντως, το πρόβλημα μπορεί να 'λυθεί' 11.3 (χρησιμοποιώντας τους βραχυκυκλωτήρες) για να κοπεί το μέγεθος σε 33.8 GB. Σε πολλές περιπτώσεις μια 4.2 (αναβάθμιση του BIOS) θα χρειαστεί αν θέλετε να εκκινήσετε το σύστημα από αυτό το δίσκο.

13 Εκτεταμένες και λογικές κατατμήσεις

6 (Παραπάνω,) είδαμε ότι τη δομή του MBR (τομέας 0): κώδικας του φορτωτή ακολουθούμενος από 4 εγγραφές κατατμήσεων 16 byte η κάθε μία, ακολουθούμενο από το AA55 αποτύπωμα. Κατατμήσεις τύπου 5 ή F ή 85 (δεκαεξαδικό) έχουν ειδική σημασία: περιγράφουν *εκτεταμένες* κατατμήσεις: κομμάτια του δίσκου που θα

κατατμηθούν σε *λογικές* κατατμήσεις. (Έτσι, μια εκτεταμένη κατάτμηση είναι απλά ένα δοχείο, δε μπορεί να χρησιμοποιηθεί από μόνη της, αλλά μέσω των λογικών κατατμήσεων που περιέχει.) Μόνο η τοποθεσία του πρώτου τομέα μιας εκτεταμένης κατάτμησης είναι σημαντική. Αυτός ο πρώτος τομέας περιέχει έναν πίνακα κατατμήσεων με 4 εγγραφές: μια λογική, μια εκτεταμένη και δυο αχρησιμοποίητες. Με αυτόν τον τρόπο μπορεί να δημιουργηθεί μια αλυσίδα πινάκων σκορπισμένη παντού στον δίσκο, όπου ο πρώτος πίνακας περιγράφει τρεις πρωταρχικές κατατμήσεις και μια εκτεταμένη, και κάθε επόμενος πίνακας περιγράφει μια λογική κατάτμηση και τον τομέα του επόμενου πίνακα.

Είναι σημαντικό να το καταλάβετε αυτό: Όταν κάποιος χρήστης κάνει κάποια βλακεία χωρίζοντας τον δίσκο του, θέλει να ξέρει: Είναι τα δεδομένα μου ακόμα εκεί; Η απάντηση είναι συνήθως ναι. Αλλά αν δημιουργήθηκαν λογικές κατατμήσεις, τότε οι πίνακες που περιγράφουν τις κατατμήσεις αυτές γράφτηκαν στην αρχή των κατατμήσεων αυτών και τα δεδομένα που υπήρχαν εκεί χάθηκαν.

Το sfdisk θα δείξει όλη την αλυσίδα. π.χ.,

```
# sfdisk -l -x /dev/hda

Disk /dev/hda: 16 heads, 63 sectors, 33483 cylinders
Units = cylinders of 516096 bytes, blocks of 1024 bytes, counting from 0

   Device Boot Start     End   #cyls   #blocks   Id  System
/dev/hda1          0+    101    102-    51376+  83  Linux
/dev/hda2        102    2133    2032   1024128  83  Linux
/dev/hda3       2134   33482   31349  15799896   5  Extended
/dev/hda4          0       -       0         0   0  Empty

/dev/hda5       2134+   6197    4064-  2048224+ 83  Linux
    -           6198   10261    4064   2048256   5  Extended
    -           2134    2133       0         0   0  Empty
    -           2134    2133       0         0   0  Empty

/dev/hda6       6198+  10261    4064-  2048224+ 83  Linux
    -          10262   16357    6096   3072384   5  Extended
    -           6198    6197       0         0   0  Empty
    -           6198    6197       0         0   0  Empty
...
/dev/hda10     30581+  33482    2902-  1462576+ 83  Linux
    -          30581   30580       0         0   0  Empty
    -          30581   30580       0         0   0  Empty
    -          30581   30580       0         0   0  Empty

#
```

Είναι δυνατόν να δημιουργηθούν χαλασμένοι πίνακες. Πολλοί πυρήνες μπαίνουν σε κύκλο αν κάποια εκτεταμένη κατάτμηση δείχνει πίσω στον εαυτό της ή σε προηγούμενη κατάτμηση στην αλυσίδα. Είναι δυνατόν να υπάρχουν δύο εκτεταμένες κατατμήσεις σε κάποιον πίνακα ώστε η αλυσίδα να χωρίζει. (Για παράδειγμα, αυτό μπορεί να συμβεί αν ένα fdisk δεν αναγνωρίζει τα 5, F, 85 ως εκτεταμένους τύπους και δημιουργήσει ένα 5 δίπλα σε ένα F.) Κανένα κοινό fdisk δε μπορεί να χειριστεί τέτοιες καταστάσεις, οπότε χρειάζεται λίγη δουλειά με το χέρι για να διορθωθούν. Ο πυρήνας του Linux θα δεχτεί μια χωρισμένη αλυσίδα σε εξωτερικό επίπεδο. Δηλαδή, μπορείτε να έχετε δυο αλυσίδες για λογικές κατατμήσεις. Μερικές φορές αυτό είναι χρήσιμο, αφού μπορείτε να έχετε τύπο 5 για το DOS και τύπο 85, αόρατο στο DOS, για το Linux, ώστε το fdisk του DOS να μην κολλήσει επειδή οι κατατμήσεις σας είναι πέραν των 1024 κυλίνδρων.

14 Λύνοντας προβλήματα

Πολλοί νομίζουν ότι έχουν πρόβλημα, ενώ στην πραγματικότητα δεν υπάρχει κανένα. Ή πιστεύουν ότι τα προβλήματα που έχουν οφείλονται στη γεωμετρία του δίσκου, ενώ αυτό δεν έχει να κάνει τίποτα με το πρόβλημά τους. Τα παραπάνω μπορεί να ακούγονται περίπλοκα αλλά η γεωμετρία είναι κάτι σχετικά απλό: αφήστε την όπως είναι και όλα θα είναι μια χαρά• ή το πολύ να δώσετε την παράμετρο 'linear' στο LILO αν δεν προχωράει μετά το 'LI' όταν ξεκινά. Προσέξτε τα μυνήματα του πυρήνα και θυμηθείτε: όσο περισσότερο πειράζετε την γεωμετρία (ορίζοντας κεφαλές και κυλίνδρους στο LILO, το fdisk και τον πυρήνα) τόσο πιθανότερο είναι να μη δουλέψει. Χοντρικά όλα είναι εντάξει από μόνα τους.

Και θυμηθείτε: πουθενά στο Linux δε χρησιμοποιείτε η γεωμετρία, οπότε κανένα πρόβλημα δε μπορεί να δημιουργηθεί από αυτήν. Η γεωμετρία χρησιμοποιείτε μόνο από το LILO και το fdisk. Έτσι, αν το LILO δεν ξεκινά τον πυρήνα, μπορεί να είναι πρόβλημα γεωμετρίας. Αν διαφορετικά λειτουργικά δεν καταλαβαίνουν τον πίνακα κατατμήσεων, μπορεί να είναι πρόβλημα γεωμετρίας. Τίποτα περισσότερο. Συγκεκριμένα, αν το mount δε δουλεύει μην ανησυχείτε για τη γεωμετρία• το πρόβλημα είναι αλλού.

14.1 Πρόβλημα: το Linux χρησιμοποιεί λάθος γεωμετρία για τον δίσκο μου.

Είναι πιθανό ένας δίσκος να λάβει λάθος γεωμετρία. Ο πυρήνας ρωτά το BIOS για τα hd0 και hd1 (τους δίσκους 80H και 81H σύμφωνα με το BIOS) και υποθέτει ότι είναι για τα hda και hdb. Αλλά αν εκκινείτε από SCSI, οι πρώτοι δυο δίσκοι μπορεί να είναι SCSI και ο πέμπτος δίσκος, που είναι πρώτος IDE hda δίσκος, παίρνει τη γεωμετρία του sda. Αυτό λύνεται εύκολα με την παράμετρο εκκίνησης 'hda=C,H,S' με τις κατάλληλες τιμές C, H και S, είτε κατά την εκκίνηση ή στο /etc/lilo.conf.

14.2 Πρόβλημα: Ίδιοι δίσκοι έχουν διαφορετική γεωμετρία

'Έχω δυο όμοιους δίσκους 10GB IBM. Το fdisk δίνει διαφορετικό μέγεθος στον καθένα. Δείτε:

```
# fdisk /dev/hdb
Disk /dev/hdb: 255 heads, 63 sectors, 1232 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot  Start      End   Blocks   Id  System
/dev/hdb1           1     1232  9896008+  83  Linux native
# fdisk /dev/hdd
Disk /dev/hdd: 16 heads, 63 sectors, 19650 cylinders
Units = cylinders of 1008 * 512 bytes

   Device Boot  Start      End   Blocks   Id  System
/dev/hdd1           1    19650  9903568+  83  Linux native
```

Πώς και;'

Τί συμβαίνει εδώ; Πρώτα από όλα, οι δίσκοι είναι πράγματι 10GB: το hdb έχει μέγεθος 255*63*1232*512 = 10133544960, και το hdd 16*63*19650*512 = 10141286400, έτσι τίποτα δε συμβαίνει και ο πυρήνας τους βλέπει και τους δύο σαν 10.1 GB. Γιατί η διαφορά στο μέγεθος; Αυτό συμβαίνει επειδή ο πυρήνας παίρνει τις πληροφορίες για τους πρώτους δυο IDE δίσκους από το BIOS, και το BIOS έχει βάλει στον hdb 255 κεφαλές (και 16*19650/255=1232 cylinders). Η στρογγυλοποίηση κοστίζει σχεδόν 8MB.

Αν θέλετε μπορείτε να αλλάξετε τον hdd με τον ίδιο τρόπο και να δώσετε στον πυρήνα κατά την εκκίνηση 'hdd=1232,255,63'.

14.3 Πρόβλημα: το fdisk βλέπει περισσότερο χώρο από το df

Το fdisk θα σας πει πόσα τεμάχια (blocks) υπάρχουν στον δίσκο. Αν δημιουργήσετε σύστημα αρχείων, π.χ. με το mke2fs, τότε το σύστημα κρατά λίγο χώρο για λόγους διαχείρισης• περίπου 4% του συνολικού χώρου, περισσότερο αν θέλετε πολλά inodes. Για παράδειγμα:

```
# sfdisk -s /dev/hda9
4095976
# mke2fs -i 1024 /dev/hda9
mke2fs 1.12, 9-Jul-98 for EXT2 FS 0.5b, 95/08/09
...
204798 blocks (5.00%) reserved for the super user
...
# mount /dev/hda9 /somewhere
# df /somewhere
Filesystem         1024-blocks  Used Available Capacity Mounted on
/dev/hda9            3574475      13  3369664      0%   /mnt
# df -i /somewhere
Filesystem           Inodes   IUsed   IFree  %IUsed Mounted on
/dev/hda9            4096000      11 4095989     0%  /mnt
#
```

Έχουμε μια κατάτμηση 4095976 blocks, κάνουμε ένα ext2 σύστημα, το προσαρτούμε και τελικά έχουμε μόνο 3574475 blocks• 521501 blocks (12%) χάθηκαν στα inodes και τη διαχείριση. Σημειώστε ότι η διαφορά μεταξύ των 3574475 και των 3369664 blocks είναι τα 13 που χρησιμοποιούνται και τα 204798 που κρατήθηκαν για τον υπερχρήστη. Το τελευταίο νούμερο μπορεί να αλλάξει με το tune2fs. Το '-i 1024' είναι λογικό να χρησιμοποιηθεί μόνο για κατατμήσεις νέων ή ταχυδρομείου, όπου υπάρχουν πολλά και μικρά αρχεία. Το συνηθισμένο θα ήταν:

```
# mke2fs /dev/hda9
# mount /dev/hda9 /somewhere
# df /somewhere
Filesystem         1024-blocks  Used Available Capacity Mounted on
/dev/hda9            3958475      13  3753664      0%   /mnt
# df -i /somewhere
Filesystem           Inodes   IUsed   IFree  %IUsed Mounted on
/dev/hda9            1024000      11 1023989     0%  /mnt
#
```

Τώρα μόνο 137501 blocks (3.3%) χρησιμοποιούνται για inodes, ώστε έχουμε 384 MB περισσότερα από πριν. (Προφανώς κάθε inode χρησιμοποιεί 128 bytes). Από την άλλη, μπορούμε να έχουμε το πολύ 1024000 αρχεία (παραπάνω από αρκετά), αντί των 4096000 (υπερβολικά πολλά) που είχαμε πριν.