NAME

cgdisk – Curses-based GUID partition table (GPT) manipulator

SYNOPSIS

cgdisk [–a] *device*

DESCRIPTION

GPT fdisk is a text–mode family of programs for creation and manipulation of partition tables. The **cgdisk** member of this family employs a curses-based user interface for interaction using a text–mode menuing system. It will automatically convert an old–style Master Boot Record (MBR) partition table or BSD disklabel stored without an MBR carrier partition to the newer Globally Unique Identifier (GUID) Partition Table (GPT) format, or will load a GUID partition table. Other members of this program family are **gdisk** (the most feature-rich program of the group, with a non-curses-based interactive user interface) and **sgdisk** (which is driven via command-line options for use by experts or in scripts). FixParts is a related program for fixing a limited set of problems with MBR disks.

For information on MBR vs. GPT, as well as GPT terminology and structure, see the extended GPT fdisk documentation at *http://www.rodsbooks.com/gdisk/* or consult Wikipedia.

The **cgdisk** program employs a user interface similar to that of Linux's **cfdisk**, but **cgdisk** modifies GPT partitions. It also has the capability of transforming MBR partitions or BSD disklabels into GPT partitions. Like the original **cfdisk** program, **cgdisk** does not modify disk structures until you explicitly write them to disk, so if you make a mistake, you can exit from the program with the Quit option to leave your partitions unmodified.

Ordinarily, **cgdisk** operates on disk device files, such as */dev/sda* or */dev/hda* under Linux, */dev/disk0* under Mac OS X, or */dev/ad0* or */dev/da0* under FreeBSD. The program can also operate on disk image files, which can be either copies of whole disks (made with **dd**, for instance) or raw disk images used by emulators such as QEMU or VMWare. Note that only *raw* disk images are supported; **cgdisk** cannot work on compressed or other advanced disk image formats.

Upon start, **cgdisk** attempts to identify the partition type in use on the disk. If it finds valid GPT data, **cgdisk** will use it. If **cgdisk** finds a valid MBR or BSD disklabel but no GPT data, it will attempt to convert the MBR or disklabel into GPT form. (BSD disklabels are likely to have unusable first and/or final partitions because they overlap with the GPT data structures, though.) Upon exiting with the 'w' option, **cgdisk** replaces the MBR or disklabel with a GPT. *This action is potentially dangerous!* Your system may become unbootable, and partition type codes may become corrupted if the disk uses unrecognized type codes. Boot problems are particularly likely if you're multi–booting with any GPT–unaware OS. If you mistakenly launch **cgdisk** on an MBR disk, you can safely exit the program without making any changes by using the Quit option.

When creating a fresh partition table, certain considerations may be in order:

* For data (non–boot) disks, and for boot disks used on BIOS–based computers with GRUB as the boot loader, partitions may be created in whatever order and in whatever sizes are desired.

* Boot disks for EFI–based systems require an *EFI System Partition* (GPT fdisk internal code 0xEF00) formatted as FAT–32. The recommended size of this partition is between 100 and 300 MiB. Boot–related files are stored here. (Note that GNU Parted identifies such partitions as having the "boot flag" set.)

* The GRUB 2 boot loader for BIOS–based systems makes use of a *BIOS Boot Partition* (GPT fdisk internal code 0xEF02), in which the secondary boot loader is stored, without the benefit of a

filesystem. This partition can typically be quite small (roughly 32 KiB to 1 MiB), but you should consult your boot loader documentation for details.

* If Windows is to boot from a GPT disk, a partition of type *Microsoft Reserved* (GPT fdisk internal code 0x0C01) is recommended. This partition should be about 128 MiB in size. It ordinarily follows the EFI System Partition and immediately precedes the Windows data partitions. (Note that old versions of GNU Parted create all FAT partitions as this type, which actually makes the partition unusable for normal file storage in both Windows and Mac OS X.)

* Some OSes' GPT utilities create some blank space (typically 128 MiB) after each partition. The intent is to enable future disk utilities to use this space. Such free space is not required of GPT disks, but creating it may help in future disk maintenance. You can use GPT fdisk's relative partition positioning option (specifying the starting sector as '+128M', for instance) to simplify creating such gaps.

OPTIONS

Only one command-line option is accepted, aside from the device filename: *−a*. This option alters the highlighting of partitions and blocks of free space: Instead of using ncurses, when *−a* is used **cgdisk** uses a ">" symbol to the left of the selected partition or free space. This option is intended for use on limited display devices such as teletypes and screen readers.

Interactions with **cgdisk** occur with its interactive text−mode menus. The display is broken into two interactive parts:

* The partition display area, in which partitions and gaps between them (marked as "free space") are summarized.

* The option selection area, in which buttons for the main options appear.

In addition, the top of the display shows the program's name and version number, the device filename associated with the disk, and the disk's size in both sectors and IEEE-1541 units (GiB, TiB, and so on).

You can use the following keys to move among the various options and to select among them:

up arrow
This key moves the partition selection up by one partition.

down arrow
This key moves the partition selection down by one partition.

Page Up
This key moves the partition selection up by one screen.

Page Down
This key moves the partition selection down by one screen.

right arrow
This key moves the option selection to the right by one item.

left arrow
This key moves the option selection to the left by one item.

Enter This key activates the currently selected option. You can also activate an option by typing the capitalized letter in the option's name on the keyboard, such as **a** to activate the Align option.

If more partitions exist than can be displayed in one screen, you can scroll between screens using the partition selection keys, much as in a text editor.

Available options are as described below. (Note that **cgdisk** provides a much more limited set of options than its sibling **gdisk**. If you need to perform partition table recovery, hybrid MBR modifcation, or other advanced operations, you should consult the **gdisk** documentation.)

Align Change the sector alignment value. Disks with more logical sectors than physical sectors (such as modern Advanced Format drives), some RAID configurations, and many SSD devices, can suffer performance problems if partitions are not aligned properly for their internal data structures. On new disks, GPT fdisk attempts to align partitions on 1MiB boundaries (2048−sectors on disks with 512-byte sectors) by default, which optimizes performance for all of these disk types. On pre−partitioned disks, GPT fdisk attempts to identify the alignment value used on that disk, but will set 8-sector alignment on disks larger than 300 GB even if lesser alignment values are detected. In either case, it can be changed by using this option.

Backup
Save partition data to a backup file. You can back up your current in−memory partition table to a disk file using this option. The resulting file is a binary file consisting of the protective MBR, the main GPT header, the backup GPT header, and one copy of the partition table, in that order. Note that the backup is of the current in−memory data structures, so if you launch the program, make changes, and then use this option, the backup will reflect your changes.

Delete Delete a partition. This action deletes the entry from the partition table but does not disturb the data within the sectors originally allocated to the partition on the disk. If a corresponding hybrid MBR partition exists, **gdisk** deletes it, as well, and expands any adjacent 0xEE (EFI GPT) MBR protective partition to fill the new free space.

Help Print brief descriptions of all the options.

Info Show detailed partition information. The summary information shown in the partition display area necessarily omits many details, such as the partitions' unique GUIDs and the partitions' sector-exact start and end points. The Info option displays this information for a single partition.

Load Load partition data from a backup file. This option is the reverse of the Backup option. Note that restoring partition data from anything but the original disk is not recommended.

naMe Change the GPT name of a partition. This name is encoded as a UTF−16 string, but proper entry and display of anything beyond basic ASCII values requires suitable locale and font support. For the most part, Linux ignores the partition name, but it may be important in some OSes. GPT fdisk sets a default name based on the partition type code. Note that the GPT partition name is different from the filesystem name, which is encoded in the filesystem's data structures. Note also that to activate this item by typing its alphabetic equivalent, you must use **M**, not the more obvious **N**, because the latter is used by the next option....

New Create a new partition. You enter a starting sector, a size, a type code, and a name. The start sector can be specified in absolute terms as a sector number or as a position measured in kibibytes (K), mebibytes (M), gibibytes (G), tebibytes (T), or pebibytes (P); for instance, **40M** specifies a position 40MiB from the start of the disk. You can specify locations relative to the start or end of the specified default range by preceding the number by a '+' symbol, as in **+2G** to specify a point 2GiB after the default start sector. The size value can use the K, M, G, T, and P suffixes, too. Pressing the Enter key with no input specifies the default value, which is the start of the largest available block for the start sector and the full available size for the size.

Quit Quit from the program *without saving your changes*. Use this option if you just wanted to view information or if you make a mistake and want to back out of all your changes.

Type Change a single partition's type code. You enter the type code using a two−byte hexadecimal number. You may also enter a GUID directly, if you have one and **cgdisk** doesn't know it. If you don't know the type code for your partition, you can type **L** to see a list of known type codes. The type code list may optionally be filtered by a search string; for instance, entering **Linux** shows only partition type codes with descriptions that include the string *Linux*. This search is performed case−sensitively.

Verify Verify disk. This option checks for a variety of problems, such as incorrect CRCs and mismatched main and backup data. This option does not automatically correct most problems, though; for that, you must use **gdisk**. If no problems are found, this command displays a summary of unallocated disk space.

Write Write data. Use this command to save your changes.

BUGS

Known bugs and limitations include:

* The program compiles correctly only on Linux, FreeBSD, and Mac OS X. In theory, it should compile under Windows if the Ncurses library for Windows is installed, but I have not tested this capability. Linux versions for x86−64 (64−bit), x86 (32−bit), and PowerPC (32−bit) have been tested, with the x86−64 version having seen the most testing. Under FreeBSD, 32−bit (x86) and 64−bit (x86−64) versions have been tested. Only 32−bit versions for Mac OS X has been tested by the author.

* The FreeBSD version of the program can't write changes to the partition table to a disk when existing partitions on that disk are mounted. (The same problem exists with many other FreeBSD utilities, such as **gpt**, **fdisk**, and **dd**.) This limitation can be overcome by typing **sysctl kern.geom.debugflags=16** at a shell prompt.

* The program can load only up to 128 partitions (4 primary partitions and 124 logical partitions) when converting from MBR format. This limit can be raised by changing the *#define MAX_MBR_PARTS* line in the *basicmbr.h* source code file and recompiling; however, such a change will require using a larger−than−normal partition table. (The limit of 128 partitions was chosen because that number equals the 128 partitions supported by the most common partition table size.)

* Converting from MBR format sometimes fails because of insufficient space at the start or (more commonly) the end of the disk. Resizing the partition table (using the 's' option in the experts' menu in **gdisk**) can sometimes overcome this problem; however, in extreme cases it may be

necessary to resize a partition using GNU Parted or a similar tool prior to conversion with GPT fdisk.

* MBR conversions work only if the disk has correct LBA partition descriptors. These descriptors should be present on any disk over 8 GiB in size or on smaller disks partitioned with any but very ancient software.

* BSD disklabel support can create first and/or last partitions that overlap with the GPT data structures. This can sometimes be compensated by adjusting the partition table size, but in extreme cases the affected partition(s) may need to be deleted.

* Because of the highly variable nature of BSD disklabel structures, conversions from this form may be unreliable −− partitions may be dropped, converted in a way that creates overlaps with other partitions, or converted with incorrect start or end values. Use this feature with caution!

* Booting after converting an MBR or BSD disklabel disk is likely to be disrupted. Sometimes re−installing a boot loader will fix the problem, but other times you may need to switch boot loaders. Except on EFI−based platforms, Windows through at least Windows 7 doesn't support booting from GPT disks. Creating a hybrid MBR (using the 'h' option on the recovery & transformation menu in **gdisk**) or abandoning GPT in favor of MBR may be your only options in this case.

* The **cgdisk** Verify function and the partition type listing obtainable by typing *L* in the Type function (or when specifying a partition type while creating a new partition) both currently exit ncurses mode. This limitation is a minor cosmetic blemish that does not affect functionality.

AUTHORS

Primary author: Roderick W. Smith (rodsmith@rodsbooks.com)

Contributors:

* Yves Blusseau (1otnwmz02@sneakemail.com)

* David Hubbard (david.c.hubbard@gmail.com)

* Justin Maggard (justin.maggard@netgear.com)

* Dwight Schauer (dschauer@gmail.com)

* Florian Zumbiehl (florz@florz.de)

SEE ALSO

cfdisk (8), **fdisk (8)**, **gdisk (8)**, **mkfs (8)**, **parted (8)**, **sfdisk (8) sgdisk (8) fixparts (8)**

http://en.wikipedia.org/wiki/GUID_Partition_Table

http://developer.apple.com/technotes/tn2006/tn2166.html

http://www.rodsbooks.com/gdisk/

AVAILABILITY

The **cgdisk** command is part of the *GPT fdisk* package and is available from Rod Smith.