fdisk: A XXI Century Disk Partitioning Tool

Davidlohr Bueso dave@gnu.org   Petr Uzel petr.uzel@suse.cz

openSUSE Conference 2012
Prague, Czech Republic

October 21st, 2012.

Disk Partitioners

GNU Parted

- Supports many disklabels
- libparted
- inflexible

Disk Partitioners

GPT fdisk

- GPT only

Disk Partitioners

Fdisk-family

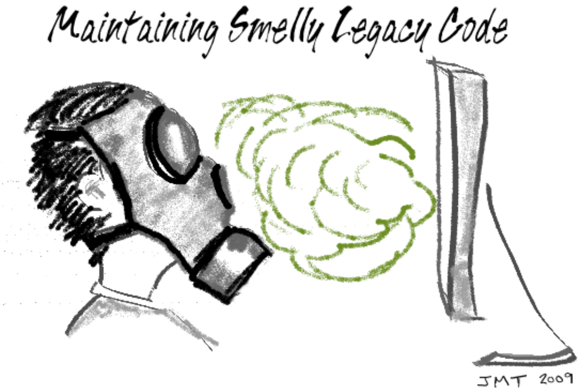
fdisk, cfdisk, sfdisk - part of util-linux

Smelly, Legacy Code

The Linux fdisk program is over 20 years old and is a complex product of multiple authors, concepts, specifications and coding styles, among others.

As a result, code is **glued** together, and making it difficult and error prone to enhance and fix bugs.

Smelly, Legacy Code



Stuck in the Past

- DOS compatibility mode
- Doesn't work with GPT
- CHS addressing
- Mainframe style UIs

Everyone Looses

Hackers loose

Adding new code and extending functionality is difficult, tedious and error prone.

Users loose

Fdisk **cannot** compete with other partitioning tools and thus looses users. Hey, healthy competition is good for everyone!

Fixing this mess

Update fdisk to modern, XXI century, disk standards.

Short Term

Short term goals:

- Cleanup and refactor current, legacy, code
- Create an internal API that abstracts disklabel concepts and specifications
- Add GUID Partition Table (GPT) support

Longer Term

Long term goals:

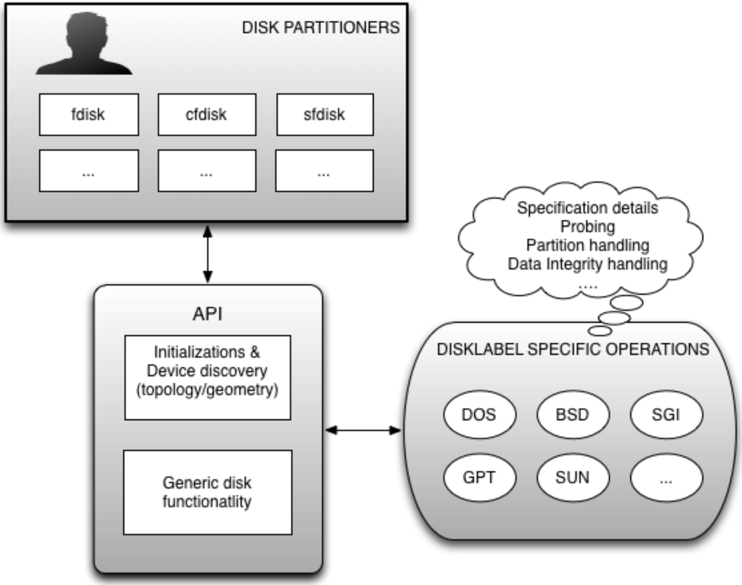
- Create an independent, libfdisk shared library.
- Rewrite cfdisk and sfdisk with new library.

Caveats

- Changes must maintain backwards compatibility.
- Write high quality code that's maintainable, at least for the next few decades.

New Internal API

- Create an abstraction level between fdisk-family tools and lower-level disklabel logic.
- Use a driver based model to deal with disklabels and handle *events* through callbacks.
- The API can be seen as:
  1. handle generic disk logic (like disk topology, sectors, MBR)
  2. gateway for disklabel specific demads (like probing or deleting a partition).
- Everything fdisk is capable of doing is goverened by a **fdisk context**.
  1. opaque data structure
  2. versioned symbols
  3. describes the disk


DISK PARTITIONERS
fdisk
cfdisk
sfdisk
...
...
...
Specification details
Probing
Partition handling
Data Integrity handling
....
API
Initializations & Device discovery (topology/geometry)
Generic disk functionatlity
DISKLABEL SPECIFIC OPERATIONS
DOS
BSD
SGI
GPT
SUN
...

API Benefits

- Unifies concepts and specifications behind different partition formats; without hiding the details.
- Simplifies dealing with disklabel specifics.
- Makes the code easier to read and modify.
- Makes detecting existing bugs easier and reduces the probability of introducing new bugs.
- Once complete, the idea is to create a shared library - similar to what libparted is to GNU parted.

What is GPT?

A standard developed by Intel in the late '90s for the layout of the partition table on a physical hard disk.

It overcomes major limitations of MBRs and today forms part of the **UEFI** standard.

Benefits of GPT

So, what's the big deal about GPT?

- Forget extended or logical DOS-like partitions. GPT can handle at least 128 primary, named, partitions.
- 64-bit addressing gives us 2^{64} available sectors, or 9.4 Zb partitions (with 512 bytes).
- 32-bit CRC checksums to ensure data integrity.
- Redundant data structures help protect against disk errors.

Drawbacks of GPT

- Compatibility
  - OS
  - Bootloaders
- Non-standard schemes (Hybrid MBRs)

Fdisk & GPT

- Well known fact that fdisk didn't play well with GPT
    - disklabel detection *only*
    - sends users to other tools (GNU parted)
    - deals only with legacy DOS partitions.
- Sept. 2012 we got full GPT support merged in mainline fdisk.

Some GPT Implementation Details

- Deals with both legacy protective and hybrid MBRs.
- Updates checksums on the fly and not only when writing in-memory data to disk.
- Plays well with larger logical-sectors (4K)
- Generous support for GUID partition types.

The Road Ahead

- Enhance UIs (*libreadline* - gdb style)
- Support more disklabels (APM, AIX)
- General cleanups and refactoring
- Documentation
- tests, tests, tests

Thank you

IT WOULD BE
TERRIBLE
IF I SLIPPED UP...
AND FAILED TO SAY
THANK
YOU!