Valgrind your filesystem

Jörn Engel

Lazybastard.org

October 30, 2009
"a method for storing and organizing computer files and the data they contain"
"a single fs/foo/ subdirectory of the linux kernel"
"a programming tool for memory debugging, memory leak detection and profiling"
Motivation

- hard to reproduce bug in logfs
- 3-4 different symptoms
- no two consecutive runs were identical
- general impression of memory corruption
Valgrind does:

- track validity and addressability
- replace libc memory allocator
- add guard memory
Valgrind can detect:

- use of uninitialized memory
- use after free
- off by one
- leaks
Problem

- valgrind only ported to Linux userspace
- some interesting filesystems run in kernelspace
Solutions

- port filesystem(s) to userspace
- port VFS to userspace
- port kernel to userspace (aka UML)

Valgrind your filesystem
Solutions

- port filesystem(s) to userspace
- **port VFS to userspace**
- port kernel to userspace (aka UML)
Externalities

- Unit tests
- Tools (fsck, dumpfs, etc.)
- Other
General approach

1. write headers (Compiler)
2. BUG() implementations (Linker)
3. real implementations (Runtime)
BUG() implementation

- saves time
- "don't write code unless you can test it"
Shortcuts

static functions

"don't export interfaces unless they are used"
copy & paste

"blurry line"
Shortcuts

- single threaded
  - (almost) no locking
  - (almost) no reference counting
  - trivial percpu counters
  - deterministic behaviour
Shortcuts

infinite memory

- GFP masks ignored
No ...

- bios
- block scheduler
- highmem
- memory alignment
- quotas
- XIP
- readahead
- namespaces
- kbuild/kconfig
fake_foo vs. sys_foo

"there was an earthquake and..."
Status

- logfs (as of April 2009) supported
- ext2 mounts, writes still buggy
Results

- several information leaks
- several memory leaks
- not the hard bug
hard bug could be deterministically reproduced...
...and no longer was a hard bug