The Linux Kernel Janitors Project

Randy Dunlap
Member of Technical Staff
SCALE 3X
Feb. 12-13, 2005

What is the Linux Kernel Janitors project?

- Started by Arnaldo Carvalho de Melo (acme@conectiva.br) to expose his large collected projects list, make it public, actually get parts of it completed (with worldwide help :)
- “TODO” list for task suggestions
- Fix bit-rotting (deprecated) code
- Fix requests from other developers
- Learn proper methods for making Linux kernel patches & working with maintainers
- Reviewed by respected, experienced Linux developers

How to get Involved

- Mailing list: kernel-janitors@lists.osdl.org (archived)
- web: http://janitor.kernelnewbies.org
- IRC: irc.kernelnewbies.org #kerneljanitor
- TODO list (but check on mailing list first)
- Also check resources at kernel newbies: http://kernelnewbies.org
- Read Documentation/ : SubmittingPatches [DCO], SubmittingDrivers, and CodingStyle
- Choose one & give it a try – you can do it

Why get involved?

- Learning curve, things are different
- Meritocracy – good ideas & code are rewarded
- Career path
- Chance to work on a real OS – any parts of it that interest you

It's Working

- ~ 250 people on the mailing list, new people subscribing often
- Some become regular contributors (“janitors”)
- I was the KJ maintainer for about 1 year and merged several hundred patches in Linux 2.5/2.6
- Second KJ maintainer is fairly new at Linux and is a student (just resigned) – just have to be willing/able to put some time in on it
- Current maintainer: 4th-year CS student, using Linux for 6 years

Mailing list etiquette & values {Reply-All}

- Linux values are basically all about technical correctness and merit, not about politics, who you know, or money
- New Linux code and patches are reviewed based on technical issues and use/misuse of the kernel API as well as (using consistent) style
- If someone attacks (flames), that's not technical, they did bad, both of you just get over it and move on.
- http://www.arm.linux.org.uk/armlinux/mletiquette.php

TODO Examples

- [Basic] Check function return codes and handle failures correctly (return resources, no leaks)
- [Basic] Reducing kernel function stack sizes
- [Medium] Removing references to discarded code sections/functions
- [Difficult] Remove use of (deprecated) functions in drivers:
  - esp. uniprocessor locking primitives

Related Projects

- kernel Makefile “check” scripts
  - make buildcheck :
    search for non-resident code/data references
  - make checkstack :
    check stack usage of all functions: kernel stack smaller, functions should use < ~512 bytes of automatic variables
  - make namespacecheck :
    search for functions that should be static (don't need to be visible globally)

Examples of Janitor Patches: Sections

- "make buildcheck" reports: Error: ./arch/i386/kernel/cpu/intel_cacheinfo.o .text refers to 0000008f R_386_32 .init.data
- Refers to this line:
  static struct _cache_table cache_table[] __initdata = ...;
- One-line patch:
  -unsigned int init_intel_cacheinfo(struct cpuinfo_x86 *c)
  +unsigned int __init init_intel_cacheinfo (struct cpuinfo_x86 *c)

Example 2: Deprecated Functions

- E.g., “check_region()” is deprecated because it is racy. Change to “request_region()”, but differerent parameters, return type, & semantics.
- On SMP: cli(), sti(), save_flags(), restore_flags(), and save_and_cli();
  see file: Documentation/cli-sti-removal.txt
- pm_register/unregister/unregister_all, send/send_all

Related Documentation

- lwn.net articles: http://lwn.net/Articles/driver-porting/
- LDD2 book: http://lwn.net/Kernel/LDD2/ (*3*)
- Driver “Dos and Dont's”: at the KJ web site
- Arjan: how not to write a driver (KJ web site)
- Greg: Coding Style, Writing Portable Code, et al (http://www.kroah.com/linux/)
- Andrew: TPP: http://www.zip.com.au/~akpm/linux/patches/stuff/tpp.txt
- Jeff: http://linux.yyz.us/patch-format.html

Kernel projects

- kernel builds of Linus & Andrew(-mm) releases, errors/warnings, sparse checks: http://developer.osdl.org/cherry/compile/
- whatever kernel area you would like to work in: drivers, MM/VM, scheduler(s), arch., security, sound, networking, ...
- Nick, Con, Adam, Robert, ...

More Related

- fix deprecated callers
- kernel bugs database: fix bugs or help update info: http://bugzilla.kernel.org or http://bugme.osdl.org
- must-fix & should-fix lists: (not up-to-date) http://www.kernel.org/pub/linux/kernel/people/akpm/must-fix/
- Trivial patch monkey (trivial@rustcorp.com.au)
  - Trivial usage rules: listed at: http://www.kernel.org/pub/linux/kernel/people/rusty/trivial/

Static source code analyzers

- Stanford Checker & Coverity: http://metacomp.stanford.edu/& http://www.coverity.com/
- sparse: http://sparse.bkbits.net or http://www.codemonkey.org.uk/projects/bitkeeper/sparse/
- smatch: http://smatch.sourceforge.net/
- Opera Research Group (UIUC)
- cqual (UC-B/UMaryland): http://www.cs.umd.edu/~jfoster/cqual/

More on source code auditing

- survey: http://www.vanheusden.com/Linux/audit.html
- Tools from the Linux secure-programming howto: http://howtos.linux.com/howtos/Secure-Programs-HOWTO/tools.shtml
- Auditing provides a path for direct or indirect (tools) kernel development

Kernel Test Projects

- LTP: http://ltp.sourceforge.net
- Open POSIX test suite: http://posixtest.sourceforge.net
- OSDL STP framework and servers: http://www.osdl.org/stp/
- UML for testing
- Virtualization for testing

About OSDL

- Founded in 2000 as a mutual benefit [501(c)6 non-profit org.]; now around 65 sponsors
- Labs in Beaverton, OR, and Tokyo, Japan
- Hardware lab access for approved projects
- Scalable Test Platform (STP) and Patch Lifecycle Manager (PLM)
- Kernel & test/performance engineering
- Working groups (initiatives)
- Linux Marketing/PR/Advocacy
- Legal

OSDL Mission

- To be the recognized center-of-gravity in the Linux industry; the central body dedicated to accelerating the use of Linux for enterprise computing through:
  - Enterprise-class testing and other technical support for the Linux development community
  - Marshalling of Linux-industry resources to focus investment on areas of greatest need thereby eliminating inhibitors to growth
  - Practical guidance to our members – vendors and end users alike – on working effectively with the Linux development community