Trust and the Linux kernel

Greg Kroah-Hartman

gregkh@linuxfoundation.org
git.sr.ht/~gregkh/presentation-linux-trust
Disclaimer

Nothing in here reflects the opinion of the Linux Foundation or any other Linux kernel developer. It’s all my personal opinion.
Open source software is more trustworthy than closed source software.
Open source software is more trustworthy than closed source software.

Because it can be audited by anyone at anytime.
Open source software is more trustworthy than closed source software.

Because it can be audited by anyone at anytime and fixed by anyone.
University of Minnesota “episode”
or
How to NOT do research on an open source community
University of Minnesota “episode”

Kernel Recipes 2022 talk
Proof that you can go back in time and audit code based on new information.
Trust
NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
Trust

“You need to verify all developers to ensure you know who they are.”
Development stats for 2021

79,662 total commits
Fixes for 2021

79,662 total commits
13,587 commits marked with Fixes: tag

17% are fixes
Fixes for 2021

79,662 total commits
13,587 commits marked with Fixes: tag

17% are fixes

Found after commits hit subsystem trees
Fixes for 2021

79,662 total commits
13,587 commits marked with Fixes: tag

17% are fixes

Found after commits hit subsystem trees

26% of the fixes were for issues before -final
2021 changes

~ 12% of all commits were fixes for problems in older releases.
2021 - Top developers

Christoph Hellwig 960 (1.2%)
Lee Jones 737 (0.9%)
Andy Shevchenko 704 (0.9%)
Mauro Carvalho Chehab 642 (0.8%)
Pavel Begunkov 624 (0.8%)
Vladimir Oltean 600 (0.8%)
Sean Christopherson 597 (0.7%)
Colin Ian King 573 (0.7%)
Arnd Bergmann 535 (0.7%)
Geert Uytterhoeven 487 (0.6%)
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dan Carpenter</td>
<td>340</td>
<td>2.5%</td>
</tr>
<tr>
<td>Arnd Bergmann</td>
<td>227</td>
<td>1.7%</td>
</tr>
<tr>
<td>Colin Ian King</td>
<td>165</td>
<td>1.2%</td>
</tr>
<tr>
<td>Sean Christopherson</td>
<td>160</td>
<td>1.2%</td>
</tr>
<tr>
<td>Vladimir Oltean</td>
<td>143</td>
<td>1.1%</td>
</tr>
<tr>
<td>Christophe JAILLET</td>
<td>142</td>
<td>1.0%</td>
</tr>
<tr>
<td>Randy Dunlap</td>
<td>140</td>
<td>1.0%</td>
</tr>
<tr>
<td>Geert Uytterhoeven</td>
<td>132</td>
<td>1.0%</td>
</tr>
<tr>
<td>Johan Hovold</td>
<td>125</td>
<td>0.9%</td>
</tr>
<tr>
<td>Eric Dumazet</td>
<td>119</td>
<td>0.9%</td>
</tr>
<tr>
<td>Authors of commits fixed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2021</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 207 (1.5%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 161 (1.1%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 121 (0.8%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 109 (0.8%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 93 (0.7%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 89 (0.6%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 77 (0.5%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 75 (0.5%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 69 (0.5%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[masqué] 67 (0.5%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authors</td>
<td>Count (Percentage)</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>207 (1.5%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>161 (1.1%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>121 (0.8%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>109 (0.8%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>93 (0.7%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>89 (0.6%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>77 (0.5%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>75 (0.5%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>69 (0.5%)</td>
<td></td>
</tr>
<tr>
<td>[redacted]</td>
<td>67 (0.5%)</td>
<td></td>
</tr>
</tbody>
</table>
Over time, the most prolific developers will write the most bugs.
Over time, the most prolific developers will write the most bugs.

So make it easy to find and fix those bugs.
Lifecycle of a kernel change
Lifecycle of a kernel change

- Submit through email
Lifecycle of a kernel change

- Submit through email
  - Development is done in public
  - No central authority (i.e. login)
  - Can be verified
  - Lowest possible barrier to entry
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
  - All through email
  - Patchwork instances to check status
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
  - Properly document your changes
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
- Average change takes 3 attempts
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
- Average change takes 3 attempts
  - Some less (1), some more (25+)
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
- Average change takes 3 attempts
- Every email submission is tested by bots
Lifecycle of a kernel change

- Submit through email
- Change is reviewed, and rejected
- Resubmit through email
- Average change takes 3 attempts
- Every email submission is tested by bots
  - and often automatically rejected
Hi Johan,

I love your patch! Perhaps something to improve:

[auto build test WARNING on usb/usb-testing]
[also build test WARNING on linus/master next-20220804]
[cannot apply to robh/for-next v5.19]

If your patch is applied to the wrong git tree, kindly drop us a note.
And when submitting patch, we suggest to use `--base` as documented in
https://git-scm.com/docs/git-format-patch

base:   https://git.kernel.org/pub/scm/linux/kernel/git/gregkh/usb.git usb-testing
config: arc-randconfig-r002-20220804 (https://download.01.org/0day-ci/archive/20220805/202208050544.ijUhoUyB-lkp@intel.com/config)
compiler: arc-elf-gcc (GCC) 12.1.0
reproduce (this is a W=1 build):

If you fix the issue, kindly add following tag where applicable
Reported-by: kernel test robot <lkp@intel.com>

All warnings (new ones prefixed by `>>`):
0-day CI Kernel Test Service  https://01.org/lkp

- Provides a one-hour response time
- Performs patch-by-patch tests
- Covers all branches of a developer tree
- Performs kernel build and static semantics-level testing
- Performs boot tests, functional, and performance tests
- Bisects code automatically when tests fail
0-day CI Kernel Test Service – benchmarks

- Virtual memory management
- I/O subsystem
- Process scheduler
- File system
- Network
- Device drivers
Lifecycle of a kernel change (cont.)

- Change accepted by maintainer
Lifecycle of a kernel change (cont.)

- Change accepted by maintainer
- Now the real testing starts
KernelCI  https://kernelci.org/

- Community-led test system focused on the upstream Linux kernel.
- Follows the open testing philosophy to enable the same collaboration to happen with testing as open source does to the code itself.
KernelCI Dashboard

- Tracks 62 different branches
- 13000+ different build/boot tests
- Thousands of different devices/architectures
- Different labs contribute from around the world

https://linux.kernelci.org/
KernelCI Dashboard  
https://linux.kernelci.org/

- Common reporting framework (KCIDB)
- Allows any kernel testing system to submit results
- Already used by RedHat and Google test labs
- One unified location to see testing results
lkft (Linux Kernel Functional Testing)
https://qa-reports.linaro.org/lkft/

- Sponsored by Linaro member companies
- Testing for stable and Linus’s -rc releases
- Testing for linux-next
- Run by tuxsuite
- 125000+ tests for modern stable releases
- Subset of arches and configs and compilers
Guenter Roeck’s test system

- Kernel maintainer doing it on their own time
- All supported kernel arches test-built (150+)
- Many build/boot tests for qemu targets (480+)
- Invaluable for stable kernel -rc testing
- Also tests linux-next and Linus’s tree
Testing every release

- kernelci
- lkft
- Guenter
- Shuah
- Android
- Huawei
- Nvidia
- Debian
- Fedora
- Many others
Trust in Linux kernel development

Trust but verify.
Trust in Linux kernel development

Trust but test.
Trust in Linux kernel development

We trust not that you will always get it right, but that you will be there to fix it when you get it wrong.